**МИНИСТЕРСТВО ЦИФРОВОГО РАЗВИТИЯ,**

**СВЯЗИ И МАССОВЫХ КОММУНИКАЦИЙ РОССИЙСКОЙ ФЕДЕРАЦИИ**

**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ**

**УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ**

**«САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ**

**ТЕЛЕКОММУНИКАЦИЙ ИМ. ПРОФ. М.А. БОНЧ-БРУЕВИЧА»**

**(СПбГУТ)**

**Санкт-Петербургский колледж телекоммуникаций им Э.Т. Кренкеля**

**Отчёт о выполнении лабораторной работы №4**

**по дисциплине "Системное программирование"**

**Тема:** Структурное программирование.

Размещение структурированных типов данных в памяти.

Принял: преподаватель Кривоносова Н.В

Выполнил: студенты группы ЗФ-053/054

Фомина Е.А.

Ткачук А.В

Шилаковски М.Э

Семененков К.И

Санкт-Петербург

2022 год

**Структурное программирование.**

**Размещение структурированных типов данных в памяти.**

1. **Цель работы**

Изучить технику использования элементов графического пользовательского интерфейса (Graphic User Interface, GUI), которые применимы для ввода данных в программу и вывода текстов.

1. **Задания**
   1. **Изучить примеры программ с GUI, в которых производится ввод и вывод информации по списку, приведенному ниже**

Для каждой программы проделать следующее:

1) Открыть ехе-файл и испытать работу всех кнопок и полей ввода;

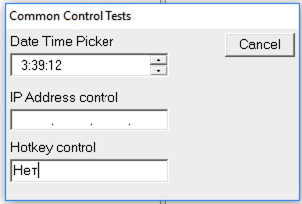
2) Сделать скриншот главного окна программы для отчета.

3) Кратко словами описать, для чего предназначена программа и в чем

особенности ее работы.

Список примеров, которые вы должны рассмотреть обязательно:

**- examples\dialogs\test;**



с классами органов управления данными и устройства выбора времени, «быстрыми клавишами» и окном с IP адресом. Данные классы автоматически валидируют пользовательский ввод на соответствие ожидаемого значения (например ЧЧ не более 24, ММ не более 60 и т.д. для времени)

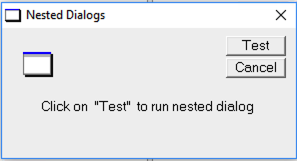
**- examples\dialogs\pbddemo;**

Изображение выглядит как стол

Автоматически созданное описание

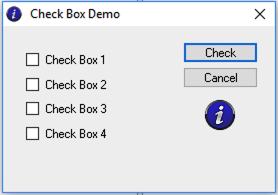
Программа предназначена для демонстрации работы с кнопками и всплывающими окнами соответствующих классов, которые валидируют пользовательский ввод на соответствие ожидаемого значения, а затем выводят полученное значение в новом окне.

**- examples\dialogs\nested;**



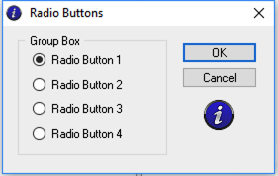
Программа предназначена для демонстрации работы с вложенными диалогами, при нажатии на кнопку «Test» открывается новое окно ввода, при заполнении которого и отправке значения выводится полученное значение в новом окне.

**- examples\dialogs\_later\check;**



Программа предназначена для демонстрации работы с чекбоксами, при выборе чекбоксов и нажатии на кнопку «Check» в новом окне выводится список состоянии чекбоксов.

**- examples\dialogs\_later\radio.**



Программа предназначена для демонстрации работы с переключателями, при выборе радиобаттона и нажатии на кнопку «OK» в новом окне выводится сообщение о выбранном значении.

**Просматривая папки с этими примерами, зафиксируйте для отчета, какие повторяющиеся типы файлов вы находите в этих папках. Сформулируйте для отчета обобщение о назначении этих файлов**

В большинстве случаев в папках с программами содержатся файлы:

– makeit.bat – пакетный файл для создания исполняемого модуля программы;

– \*.exe – исполняемый модуль программы;

– \*.asm – текст программы на МАСМ;

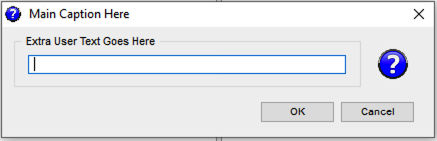
– \*.ico – файл иконки программы;

– manifest.xml – файл «манифеста»;

– rsrc.rc – ресурсный файл.

* 1. **Откройте пример examples\dialogs\_later\gettext.**

1. Проделайте с ним ту же работу, что и с примерами из п. 1.

Программа предназначена для демонстрации работы с окном ввода, при заполнении которого и отправке значения кнопкой «OK» в новом окне выводится полученное значение.

2) Скопируйте asm-файл этого примера в свою рабочую папку, чтобы при

последующем редактировании не портить исходник примера.

3) Составьте таблицу вызовов процедур и макросов для gettext.asm до

уровня процедур из MSDN. Как это делать, описано в теоретических сведениях.

**Таблица вызовов процедур и макросов для gettext.asm**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Тип | Имя пр/макро | Вызвано | Вызвано | Описано в |
| 1 | 2 | 3 | 4 | 5 |
| p | GetModuleHandle |  |  | MSDN |
| p | ExitProcess |  |  | MSDN |
| p | InitCommonControls |  |  | MSDN |
| p | LoadIcon |  |  | MSDN |
| m | GetTextDialog |  |  |  |
| m |  | Dialog |  | dialogs.inc |
| p |  |  | GlobalAlloc | MSDN |
| m |  |  | ustring | dialogs.inc |
| p |  | DlgIcon |  | dialogs.inc |
| p |  | DlgGroup |  | dialogs.inc |
| p |  | DlgEdit |  | dialogs.inc |
| p |  | DlgButton |  |  |
| m |  |  | ustring | dialogs.inc |
| p | GlobalFree |  |  | MSDN |

4) Подробно (пооператорно) прокомментируйте копию исходника

gettext.asm, которую вы в п. 2) скопировали в свою рабочую папку.

; ��������������������������������������������������������������������������������������������������  
; This example does not use the common control library as it only uses the standard controls  
; ��������������������������������������������������������������������������������������������������  
  
 include \masm32\include\masm32rt.inc ;подключение библиотеки встроенных функций masm32 (MASM32 Runtime Library)  
 include \masm32\include\dialogs.inc ;избыточное подключение макроса шаблонов диалога (вызов присутствует в masm32rt.inc)   
  
 dlgproc PROTO :DWORD,:DWORD,:DWORD,:DWORD ;протитип процедуры  
 GetTextDialog PROTO :DWORD,:DWORD,:DWORD ;прототип процедуры  
  
 .data?  
 hInstance dd ? ;объявление глобальной переменной для указателя на модуль  
  
 .code ;начало секции кода  
  
; ��������������������������������������������������������������������������������������������������  
  
start: ;метка start  
 mov hInstance, rv(GetModuleHandle,NULL) ;инициируем hInstance  
 call main ;вызов главной функции  
 invoke ExitProcess,eax ;вызов API функции ExitProcess  
  
; ��������������������������������������������������������������������������������������������������  
  
main proc ;начало главной функции  
  
 LOCAL ptxt :DWORD ;объявление локальной переменной ptxt  
 LOCAL hIcon :DWORD ;объявление локальной переменной hIcon  
  
 invoke InitCommonControls ;вызов API функции InitCommonControls  
  
 mov hIcon, rv(LoadIcon,hInstance,10) ;загрузка иконки  
  
 mov ptxt, rv(GetTextDialog," Main Caption Here"," Extra User Text Goes Here ",hIcon) ;обращение к функции отображения окна ввода   
  
 .if ptxt != 0 ;проверка наличия пользовательских данных  
 fn MessageBox,0,ptxt,"Title",MB\_OK ;вывод всплывающего окна с введенными данными  
 .else  
 fn MessageBox,0,"Cancel was pressed","Title",MB\_OK ;вывод всплывающего окна с сообщением о нажатии кнопки Cancel  
 .endif  
  
 invoke GlobalFree,ptxt ;вызов API функции GlobalFree  
  
 ret ;выход из процедуры  
  
main endp ;конец главной функции  
  
; ��������������������������������������������������������������������������������������������������  
  
GetTextDialog proc dgltxt:DWORD,grptxt:DWORD,iconID:DWORD ;функция получения пользовательского ввода  
  
 LOCAL arg1[4]:DWORD  
 LOCAL parg :DWORD  
  
 lea eax, arg1  
 mov parg, eax  
  
 ; ---------------------------------------  
 ; load the array with the stack arguments  
 ; ---------------------------------------  
 mov ecx, dgltxt  
 mov [eax], ecx  
 mov ecx, grptxt  
 mov [eax+4], ecx  
 mov ecx, iconID  
 mov [eax+8], ecx  
  
 Dialog "Get User Text", \ ; caption  
 "Arial",8, \ ; font,pointsize  
 WS\_OVERLAPPED or \ ; styles for  
 WS\_SYSMENU or DS\_CENTER, \ ; dialog window  
 5, \ ; number of controls  
 50,50,292,80, \ ; x y co-ordinates  
 4096 ; memory buffer size  
  
 DlgIcon 0,250,12,299  
 DlgGroup 0,8,4,231,31,300  
 DlgEdit ES\_LEFT or WS\_BORDER or WS\_TABSTOP,17,16,212,11,301  
 DlgButton "OK",WS\_TABSTOP,172,42,50,13,IDOK  
 DlgButton "Cancel",WS\_TABSTOP,225,42,50,13,IDCANCEL  
  
 CallModalDialog hInstance,0,dlgproc,parg  
  
 ret  
  
GetTextDialog endp  
  
; ��������������������������������������������������������������������������������������������������  
  
dlgproc proc hWin:DWORD,uMsg:DWORD,wParam:DWORD,lParam:DWORD ; функция отображения диалогового окна  
  
 LOCAL tlen :DWORD  
 LOCAL hMem :DWORD  
 LOCAL hIcon :DWORD  
  
 switch uMsg  
 case WM\_INITDIALOG  
 ; -------------------------------------------------  
 ; get the arguments from the array passed in lParam  
 ; -------------------------------------------------  
 push esi  
 mov esi, lParam  
 fn SetWindowText,hWin,[esi] ; title text address  
 fn SetWindowText,rv(GetDlgItem,hWin,300),[esi+4] ; groupbox text address  
 mov eax, [esi+8] ; icon handle  
 .if eax == 0  
 mov hIcon, rv(LoadIcon,NULL,IDI\_ASTERISK) ; use default system icon  
 .else  
 mov hIcon, eax ; load user icon  
 .endif  
 pop esi  
  
 fn SendMessage,hWin,WM\_SETICON,1,hIcon  
 invoke SendMessage,rv(GetDlgItem,hWin,299),STM\_SETIMAGE,IMAGE\_ICON,hIcon  
 xor eax, eax  
 ret  
  
 case WM\_COMMAND  
 switch wParam  
 case IDOK  
 mov tlen, rv(GetWindowTextLength,rv(GetDlgItem,hWin,301))  
 .if tlen == 0  
 invoke SetFocus,rv(GetDlgItem,hWin,301)  
 ret  
 .endif  
 add tlen, 1  
 mov hMem, alloc(tlen)  
 fn GetWindowText,rv(GetDlgItem,hWin,301),hMem,tlen  
 invoke EndDialog,hWin,hMem  
 case IDCANCEL  
 invoke EndDialog,hWin,0  
 endsw  
 case WM\_CLOSE  
 invoke EndDialog,hWin,0  
 endsw  
  
 xor eax, eax  
 ret  
  
dlgproc endp  
  
; ��������������������������������������������������������������������������������������������������  
  
end start ;окончание start (точки входа в программу)

* 1. Сделать вывод по работе, в котором описать на основании проделанной здесьработы, что, по-вашему, должно быть сделано в GUI-программе для ввода в программутекстовой строки с клавиатуры и вывода текстовой строки на экран.

1. **Контрольные вопросы.**
   1. **Что означает понятие «GUI»?**

GUI (Graphic User Interface) – Графический интерфейс пользователя - предполагает взаимодействие пользователя с компьютером при помощи разнообразных графических элементов управления, например: окон, иконок, меню, указателей.

* 1. **Зачем бывает нужен файл manifest.xml?**

Манифест (manifest) - это текст на языке XML, описывающий программу и графические библиотеки, необходимые для ее работы. До выхода Windows XP часть оконных классов - так называемые базовые классы окон - кнопки, статические тексты (static text), поля ввода (editbox), скроллбары, списки и комбобоксы - хранилась в основной библиотеке (версия 5.80), отвечающей за элементы пользовательского интерфейса - user.exe. А вот в Windows XP новую версию библиотеки этих классов (версия 6.0) поместили в библиотеку comctl32.dll.

Таким образом обеспечена обратная совместимость со старыми программами. Но чтобы сообщить программе, какую из двух библиотек следует использовать, и нужен манифест

* 1. **В OS Windows кто исполняет вызовы процедур, описанных в MSDN?**

В ОС Windows между приложением и совокупностью системных вызовов (системных сервисов в терминологии Microsoft) расположен дополнительный абстрактный слой - программный интерфейс Win32 API. За счет этого Win32-приложение может работать практически во всех версиях Windows несмотря на то, что сами системные вызовы в различных версиях системы различны и не документированы.