Министерство образования Российской Федерации

**Комсомольский-на-Амуре государственный технический университет**

**Факультет компьютерных технологий**

**кафедра «Математического обеспечения и применения ЭВМ»**

**Тихомиров В.А.**

**ПРОГРАММНЫЕ ФУНКЦИИ**

**ОПЕРАЦИОННОЙ СИСТЕМЫ**

**WINDOWS**

лабораторная работа 1

##### Комсомольск-на-Амуре

#### УДК 519.682.1

Программные функции операционной системы Windows. Лабораторная работа /Сост. Тихомиров В.А. – Комсомольск-на-Амуре: Комсомольский-на-Амуре государственный технический университет, - 12 с.

Приводятся основы программирования простейших приложений в среде Windows с использованием системных функций. Разбор особенностей функционирования приложения в операционной системе проводится на примере приложений, выполненных на языке С++. Подробно рассматривается механизм передачи параметров в системные функции Windows, совместная работа приложения и операционной системы: регистрация окон, цикл обработки сообщений и т.п. Разбираются особенности компоновки исполняемого файла в ОС Windows.

Разработка предназначена для студентов первого и второго высшего образования

1. ВВЕДЕНИЕ

Целью данной лабораторной работы является ознакомление студентов, с системными функциями ОС Windows, использованием их в приложениях на уровне операционной системы.

**2. ПРОСТЕЙШЕЕ 32-х РАЗРЯДНОЕ ПРИЛОЖЕНИЕ**

Начиная с процессора i386 появилась технология страничной адресации расширенной памяти ПЭВМ, при которой используются расширенные до 32 бит регистры процессора. Приложения, основанные на такой адресации, являются более устойчивыми в работе, более быстро выполняются, однако по размеру исполняемых файлов минимум в двое превосходят 16-ти разрядные приложения.

Создадим пример такого приложения на базе всего одной системной функции – функции выдачи системного сообщения.

Любому, даже начинающему разработчику информационных систем, должно быть известно, что все события в ОС Windows происходят в окнах.

Чтобы вывести на экран нужный нам текст следует создать какое-нибудь окно. Желательно – что-нибудь по проще. Открываем справочник системных функций (API), предоставляемых ОС Windows программисту и начинаем его изучать.

Следует сказать, что наличие справочника по API функциям для низкоуровневого программирования в Windows вещь абсолютно необходимая. Она так же важна, как необходимость при программировании в DOS иметь справочник по функциям BIOS и DOS.

Проще всего справочник по API функциям получить в электронном виде, установив на своем компьютере систему HELP SDK из пакета MSDN для VC++.

Итак, почитав несколько вечеров вышеуказанный справочник и познакомившись с основными функциями и сервисами, которые предоставляет ОС программисту, приходим к выводу, что для нашей задачи лучше всего подходит функция MessageBox. Вот что про нее написано в справочнике на чистом английском языке, который мы без труда понимаем:

## **ФУНКЦИЯ MessageBox**

(ОкноСообщения)

Функция **MessageBox** создает, отображает на экране и оперирует окном сообщений. Окно сообщений содержит определяемое программой сообщение и заголовок, плюс любую комбинацию предопределенных пиктограмм и командных кнопок.

Синтаксис

int MessageBox  
(  
HWND hWnd,// дескриптор окна владельца  
LPCTSTR lpText,// адрес текста в окне сообщений  
LPCTSTR lpCaption,// адрес заголовка в окне сообщений  
UINT uType // стиль окна сообщений  
);

Параметры  
**hWnd**  
Идентифицирует окно владельца блока сообщений, которым оно было создано. Если этот параметр имеет значение ПУСТО (**NULL**), у блока сообщения нет окна владельца.  
**lpText**  
Указывает на строку с символом нуля в конце, содержащую сообщение, которое должно быть отражено на экране.  
**lpCaption**  
Указывает на строку с символом нуля в конце, используемую для заголовка диалогового окна. Если этот параметр значение ПУСТО (**NULL**), то по умолчанию используется заголовок Ошибка (**Error**).  
**uType**  
Определяет установку битов флажков, которые обуславливают содержание и поведение диалогового окна. Этот параметр может быть комбинацией флажков из ниже следующих групп флажков.  
Определите один из следующих флажков, чтобы указать кнопки, содержащиеся в окне сообщений:

* **MB\_ABORTRETRYIGNORE** - Окно сообщение содержит три командных кнопки: Прервать (Abort), Повторить (Retry) и Проигнорировать (Ignore).
* **MB\_OK** - Окно сообщение содержит одну командную кнопку: OK. Это по умолчанию.
* **MB\_OKCANCEL** - Окно сообщение содержит две командных кнопки: OK и Отменить (Cancel).
* **MB\_RETRYCANCEL** - Окно сообщение содержит две командных кнопки: Повторить (Retry) и Отменить (Cancel).
* **MB\_YESNO** - Окно сообщение содержит две командных кнопки: Да (Yes) и Нет (No).
* **MB\_YESNOCANCEL** - Окно сообщение содержит три командных кнопки: Да (Yes), Нет (No) и Отменить (Cancel).

Определите один из следующих флажков, чтобы отобразить пиктограмму в окне сообщений:

* **MB\_ICONEXCLAMATION, MB\_ICONWARNING** - В окне сообщений появляется пиктограмма восклицательного знака.
* **MB\_ICONINFORMATION, MB\_ICONASTERISK** - В окне сообщений появляется пиктограмма, состоящая из символа i нижнего регистра в круге.
* **MB\_ICONQUESTION** - В окне сообщений появляется пиктограмма в виде знака вопроса.
* **MB\_ICONSTOP,MB\_ICONERROR,MB\_ICONHAND** - В окне сообщений появляется пиктограмма в виде стоп-сигнала.

Определите один из следующих флажков, чтобы указать заданную по умолчанию кнопку:

* **MB\_DEFBUTTON1** - Первая кнопка - основная кнопка. MB\_DEFBUTTON1 - значение по умолчанию, если не определена кнопка MB\_DEFBUTTON2, MB\_DEFBUTTON3 или MB\_DEFBUTTON4.
* **MB\_DEFBUTTON2** - Вторая кнопка - основная кнопка.
* **MB\_DEFBUTTON3** - Третья кнопка - основная кнопка.
* **MB\_DEFBUTTON4** - Четвертая кнопка - основная кнопка.

Определите один из следующих флажков, чтобы указать модальность диалогового окна:

* **MB\_APPLMODAL** - Пользователь должен ответить окну сообщений перед продолжением работы в окне, которое идентифицировано параметром hWnd. Однако, пользователь может перемещаться в окнах других прикладных программ и работать в этих окнах.В зависимости от иерархии окон в прикладной программе, пользователь может получить возможность, чтобы перемещаться в другие окна в пределах прикладной программы. Все дочерние окна родителя окна сообщений автоматически блокируются, однако выскакивающие окна - нет.MB\_APPLMODAL - значение по умолчанию, если не определен флажок, ни MB\_SYSTEMMODAL, ни MB\_TASKMODAL.
* **MB\_SYSTEMMODAL** - То же самое, что и MB\_APPLMODAL за исключением того, что окно сообщений имеет стиль WS\_EX\_TOPMOST. Используйте системно - модальные окна сообщений, чтобы уведомлять пользователя о серьезных, потенциально опасных ошибках, которые требуют немедленного внимания (например, запуск программы при нехватке памяти). Этот флажок не имеет никакого влияния на способность пользователя взаимодействовать с другими окнами, а не те, которые связаны с hWnd.
* **MB\_TASKMODAL** - То же самое, что и MB\_APPLMODAL за исключением того, что все окна верхнего уровня, принадлежащие текущей задаче, заблокированы, если параметр hWnd имеет значение ПУСТО (NULL). Используйте этот флажок, когда вызывающая прикладная программа или библиотека не имеют доступного дескриптора окна, но все еще должны сохранять вводимые данные для других окон в текущей прикладной программе без приостановки работы других прикладных программ.

В дополнение, вы можете устанавливать ниже перечисленные флажки:  
**MB\_DEFAULT\_DESKTOP\_ONLY**  
Рабочий стол, в настоящее время принимающий ввод, должен быть заданным по умолчанию рабочим столом; иначе, функция не выполняет задачу. Заданный по умолчанию рабочий стол - первая запущенная прикладная программа, после того, как пользователь вошел в систему.  
**MB\_HELP**  
Прибавляет кнопку Справка (Help) в окно сообщений. Выбор кнопки **Help** или нажатие **F1** генерирует событие появления Справки.  
**MB\_RIGHT**  
Выравнивание текста справа.  
**MB\_RTLREADING**  
Отображает на экране сообщение и текст заголовка с использованием порядка зеркального отображения для Еврейских и Арабских систем письменности.  
**MB\_SETFOREGROUND**  
Окно сообщений становится приоритетным окном. Внутри **Windows** для окна сообщений вызывает функцию **SetForegroundWindow.**  
**MB\_TOPMOST**  
Окно сообщений создается со стилем окна **WS\_EX\_TOPMOST**.  
**MB\_SERVICE\_NOTIFICATION**  
Только для **Windows NT:** вызывающая программа является обслуживающей по уведомлению пользователя о событии. Функция отображает окно сообщений на текущем активном рабочем столе, даже если никто из пользователей не вошел в систему компьютера.  
Если этот флажок установлен, параметр **hWnd** должен иметь значение ПУСТО (**NULL**). Это такое окно сообщений, которое может появляться на другом рабочем столе, а не только на том, которое соответствует **Wnd**.  
Для **Windows NT** версии 4.0, значение **MB\_SERVICE\_NOTIFICATION** изменилось. См. **WINUSER.H** для старых и новых значений. **Windows NT 4.0**обеспечивает совместимость вниз для существующих ранее услуг, при помощи преобразования данных старых значений в новых значениях при реализации **MessageBox** и **MessageBoxEx**. Это преобразование данных делается только для исполнимых программ (**.exe**), которые имеют номер версии, как установлено компоновщиком, меньше чем **4.0**.

Чтобы сформировать обслуживание, которое использует **MB\_SERVICE\_NOTIFICATION** и возможность запускать, и **Windows NT 3.x** и **Windows NT 4.0**, Вы имеете два пути:

1. Во время компоновки, определите номер версии меньше чем 4.0; или
2. Во время компоновки, определите версию 4.0.

Во время запуска, используйте функцию GetVersionEx, чтобы проверить системную версию. Тогда при продолжении запуска Windows NT 3.x, используйте MB\_SERVICE\_NOTIFICATION\_NT3X; а для Windows NT 4.0, используйте MB\_SERVICE\_NOTIFICATION.  
**MB\_SERVICE\_NOTIFICATION\_NT3X**  
Только для **Windows NT:** Это значение соответствует значению, определенному для **MB\_SERVICE\_NOTIFICATION** для версии **3.51 Windows NT**.

Возвращаемые значения  
Возвращаемое значение нулевое, если недостаточно памяти, чтобы создать окно сообщений.  
Если функция завершается успешно, возвращаемое значение - одно из следующих значений элемента меню, возвращаемых диалоговым окном:

* **IDABORT** - Была выбрана кнопка Прервать (Abort).
* **IDCANCEL** - Была выбрана кнопка Отменить (Cancel).
* **IDIGNORE** - Была выбрана кнопка Игнорировать (Ignore).
* **IDNO** - Была выбрана кнопка Нет (No).
* **IDOK** - Была выбрана кнопка OK.
* **IDRETRY** - Была выбрана кнопка Повторить (Retry).
* **IDYES** - Была выбрана кнопка Да (Ye).

Если окно сообщений имеет кнопку Отменить (**Cancel**), функция возвращает значение **IDCANCEL** тогда, если или нажата клавиша **ESC**, или выбрана кнопка **Cancel**. Если у окна сообщений нет кнопки **Cancel**, нажатие на **ESC** не имеет никакого эффекта.

Вот эту функцию мы и используем в программе для вывода нашего текста. Функция не требует длительной подготовки к своему использованию: описанию параметров и регистрации окна, в качестве окна-родителя для нее можно выбрать центральное окно ОС Windows, которое называется DeskTop. Опишем эту функцию как внешнюю (функция будет предоставлена операционной системой «из вне…») в тексте нашей программы:

Для выполнения задания, запускаем среду разработки Visual Studio. Создаем в ней ПУСТОЙ КОНСОЛЬНЫЙ проект и вписываем в него следующий текст:

#include <windows.h>

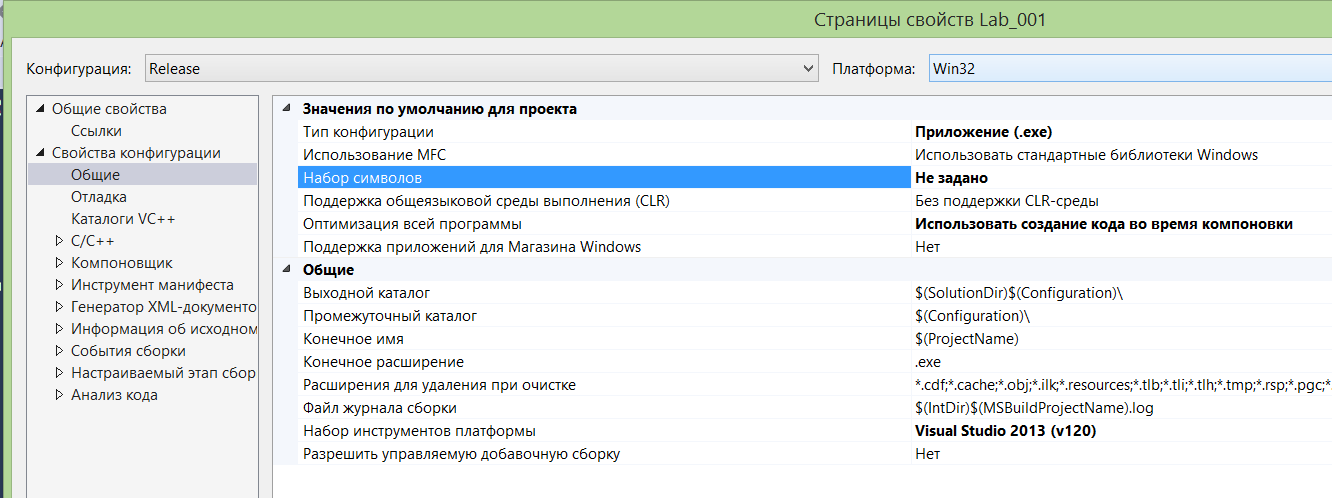
void main(void)

{

MessageBoxA(0, "Запуск ПЕРВОЙ программы", "ТЕСТ", 0);

}

Настраиваем использование шрифтов ANSI кодировки



Компилируем и запускаем приложение.

Программа должна вывести на экран окно сообщения.

Правда вместе с ним выводится и пустое консольное окно. Это не красиво.

Чтобы консольного окна не было, сделаем новый проект, но уже типа WIN 32.

Текст его модуля теперь будет:

#undef UNICODE

#include <Windows.h>

int WINAPI WinMain(HINSTANCE hInstance,   
 HINSTANCE hPrevInstance,   
 PSTR szCmdLine,   
 int iCmdShow)

{

MessageBox(0, "Запуск ПЕРВОЙ программы", "ТЕСТ", 0);

}

Изменился вид точки входа в программу.

Компилируем, запускаем.

Теперь окно сообщения чистое, без консольного окна.

ЗАДАНИЕ

1. Изучите теоретическую часть и примеры, описанные в step1\_Win32.doc методической разработке;
2. Ознакомьтесь с SDK по API функциям операционной системы Windows;
3. Реализуйте на ПЭВМ в ОС Windows пример 32-х разрядного приложения, приведенные в методической разработке step1\_Win32.doc;
4. Доработайте пример созданного приложений согласно варианта, заданного в таблице:

|  |  |
| --- | --- |
| № вар. | Задание на доработку программы |
| 1 | Добавить на выводимое сообщение три кнопки и системную пиктограмму с предупреждающим значком. При нажатии на кнопки должна меняться пиктограмма. А при нажатии на одну и ту же кнопку два раза – заканчиваться программа. |
| 2 | Поместить на выводимом сообщении две кнопки «ДА» и «НЕТ». При нажатии на кнопку «ДА» должно появляться новое сообщение, а старое - исчезать. При нажатии на кнопку «НЕТ» заканчивается работа программы. |
| 3 | Поместить на выводимом сообщении три кнопки «ДА», «НЕТ», «Отмена». При нажатии на кнопку «ДА» должно появляться новое сообщение. При нажатии на кнопку «НЕТ» - меняться пиктограмма на сообщении, при нажатии на «Отмена» - заканчивается работа программы. |
| 4 | Поместить на выводимом сообщении три кнопки «ДА», «НЕТ», «Отмена». При нажатии на кнопку «ДА» должен меняться основной текст сообщения. При нажатии на кнопку «НЕТ» - должен меняться текст заголовка окна сообщения, при нажатии на «Отмена» - заканчивается работа программы. |
| **5** | С помощью выводимых сообщений составить диалог с пользователем, в ходе которого на сообщениях появляются и используются в диалоге разные кнопки и разные пиктограммы. Сценарий диалога разработать самостоятельно. |
| **6** | Поместить на выводимом сообщении три кнопки «ДА», «НЕТ», «Отмена». При нажатии на кнопку «ДА» должно появляться новое сообщение. При нажатии на кнопку «НЕТ» - меняться пиктограмма на сообщении, при нажатии на «Отмена» - заканчивается работа программы. |
| **7** | Поместить на выводимом сообщении три кнопки «ДА», «НЕТ», «Отмена» и составить программу – тест: пользователь отвечает на вопросы «да» или «нет» и в конце ему выдается некоторый результат. При нажатии на «Отмена» - заканчивается работа программы. |
| **8** | Поместить на выводимом сообщении три кнопки «ДА», «НЕТ», «Отмена». При нажатии на кнопку «ДА» должен меняться основной текст сообщения. При нажатии на кнопку «НЕТ» - должен меняться текст заголовка окна сообщения, при нажатии на «Отмена» - заканчивается работа программы. |
| 9 | Добавить на выводимое сообщение три кнопки и системную пиктограмму с предупреждающим значком. При нажатии на кнопки должна меняться пиктограмма. А при нажатии на одну и ту же кнопку два раза – заканчиваться программа. |
| 10 | Поместить на выводимом сообщении три кнопки «ДА», «НЕТ», «Отмена». При нажатии на кнопку «ДА» должен меняться основной текст сообщения. При нажатии на кнопку «НЕТ» - должен меняться текст заголовка окна сообщения, при нажатии на «Отмена» - заканчивается работа программы. |
| **11** | Поместить на выводимом сообщении три кнопки «ДА», «НЕТ», «Отмена» и составить программу – тест: пользователь отвечает на вопросы «да» или «нет» и в конце ему выдается некоторый результат. При нажатии на «Отмена» - заканчивается работа программы. |