МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ

РОССИЙСКОЙ ФЕДЕРАЦИИ

ФГБОУ ВО

«БРЯНСКИЙ ГОСУДАРСТВЕННЫЙ

ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

**Кафедра** «Информатика и программное обеспечение»

**РАСЧЕТНО-ГРАФИЧЕСКАЯ РАБОТА**

по дисциплине

**«Программирование Windows-приложений»**

на тему:

**Работа с окнами, пользовательским интерфейсом**

Вариант №45

Выполнил студент гр. О-18-ПРИ-рпс-Б

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Подгорняк А.А.

«\_\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_2021

Руководитель

\_\_\_\_\_\_\_\_\_\_\_\_к. т. н., доц. Белов Е. А.

«\_\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_2021

Брянск 2021

**СОДЕРЖАНИЕ**

[1. Задания 3](#_Toc73294070)

[2. Теоретический материал 4](#_Toc73294071)

[2.1. Вывод графики в окно 4](#_Toc73294072)

[2.2. Стили окон 5](#_Toc73294073)

[3. Описание программ 7](#_Toc73294074)

[3.1. Задание №1 7](#_Toc73294075)

[3.2. Задание №2 8](#_Toc73294076)

[4. Контрольные примеры работы программ 11](#_Toc73294077)

[4.1. Задание №1 11](#_Toc73294078)

[4.2. Задание №2 11](#_Toc73294079)

[Список литературы 13](#_Toc73294080)

# Задания

* Задание №1. Создать два. списка, между ними расположить кнопки "Переместить»" и "«Переместить". Левый список должен быть изначально заполнен списком строк. Если выбрать строку и нажать одну из кнопок, то выбранная строка должна переместиться слева направо или наоборот.
* Задание №2. Создать диалоговую панель для задания текста выводимой строки, цвета букв, цвета фона текста и ориентации вывода. После выбора атрибутов вывода отобразить указанную строку с этими атрибутами.

# Теоретический материал

## Вывод графики в окно

Вывод графики в окна обладает рядом особенностей.

Нельзя пользоваться функциями вывода библиотеки компилятора, так как они приспособлены для вывода в одно и единственное окно. В операционной системе Windows приложения выводят одновременно в различные окна. Система сама решает все проблемы, связанные с возможным перекрытием или перемещением этих окон. С этим обстоятельством связано то, что в окно стремятся выводить в одном месте приложения – при обработке сообщения WM\_PAINT. Приложение описывают таким образом, чтобы при поступлении сообщения WM\_PAINT функция окна могла перерисовать все окно или любую его заданную часть.

Интерфейс графических устройств (Graphics Device Interface – GDI) системы открывает доступ к большому числу функций вывода. Приложения, обращаясь к функциям GDI, работают не с физическими устройствами вывода, а с логическими, т. е. описание вызова функций не зависит от физического способа отображения. Функция GDI передает указания о выводе драйверу устройства вывода. Драйвер работает непосредственно с физическим устройством и при управлении выводом учитывает его ограниченные возможности и аппаратные особенности. Благодаря этому приложения способны работать с любым устройством вывода, драйвер которого установлен в системе. Однако не все физические устройства вывода способны поддерживать те режимы, в которых работает видеомонитор, поэтому, описывая последовательность операций вывода, например, на принтер, следует учитывать ограниченные возможности установленного принтера.

Параметры вывода устанавливают в контексте отображения с помощью функций GDI. Контекст отображения – это структура данных, которая содержит характеристики устройства вывода и указатели на выбранные инструменты рисования. Функции GDI используют только выбранные в контекст отображения параметры и инструменты рисования, например, для рисования линии некоторой толщины в контекст отображения, приложение должно выбрать перо этой толщины.

Дескриптор контекста отображения служит первым аргументом вызова всех функций, связанных с выводом в окно.

GDI, с точки зрения программиста – это контекст отображения.

Контекст отображения можно сравнить с листом бумаги, на котором приложение рисует, а также инструментом для рисования. Инструменты для рисования – это перья, кисти, а также шрифты и целые графические изображения, с помощью которых приложение создаст необходимый рисунок.

Кроме контекста отображения и инструмента для рисования, приложению доступны десятки функций программного интерфейса GDI.

Контекст отображения является структурой данных, описывающих устройство отображения. В этой структуре хранятся различные характеристики устройства и набор инструментов для рисования, выбранные по умолчанию. Приложение может выбрать в контекст отображения различные инструменты (например, перья различной толщины и цвета). Поэтому, если необходимо нарисовать линию красного цвета, то перед выполнением операции рисования, необходимо выбрать в контекст отображения соответствующее перо.

Функции рисования не имеют параметров, указывающих цвет и толщину линий. Такие параметры хранятся в контексте отображения.

Приложение может создать контекст отображения не только для экрана монитора, но и для любого другою графического устройства, например, для принтера. В этом случае оно может рисовать на принтере различные изображения, используя те же функции, что и при рисовании на экране. Можно создать контекст отображения для метафайла. Метафайл – это обычный файл на диске или в памяти, в котором хранятся последовательности команд интерфейса GDI. Приложение может выполнить вывод графической информации в метафайл, как в обычное устройство вывода, а затем «проиграть» метафайл на реальном устройстве вывода.

## Стили окон

По совокупности свойств различают перекрывающиеся (overlapped), временные (pop-up) и дочерние (child) окна.

Перекрывающиеся окна чаще используют в качестве окон приложения. Они всегда имеют заголовок, рамку и рабочую область окна, могут иметь системное меню, кнопки восстановления размеров, закрытия и сворачивания окна в пиктограмму, горизонтальную и вертикальную полосы просмотра, меню, панель инструментов и строку состояния. Базовый стиль таких окон описан константой WS\_OVERLAPPED. Чаще используемый стиль окон WS\_OVERLAPPEDWINDOW в дополнение к базовому указывает, что окно имеет системное меню, кнопки восстановления размеров, закрытия и сворачивания окна.

Перекрывающееся окно может принадлежать другому окну. Если окно-владелец сворачивается в пиктограмму, то подчиненные ему окна становятся невидимыми. При уничтожении окна автоматически уничтожаются подчиненные ему окна. Подчиненные окна всегда располагаются над поверхностью окна владельца, загораживая его.

Временные окна обычно используют для вывода сообщений пользователю и остаются на экране непродолжительное время. Базовый стиль временного окна описан константой WS\_POPUP. Такое окно по умолчанию не имеет заголовка. Чаще временное окно описывают константой WS\_POPUPWINDOW. Для добавления к временному окну системного меню и заголовка стиль WS\_POPUPWINDOW комбинируют со стилем WS\_CAPTION. Во всем остальном временные окна – это специальный вид перекрывающихся окон.

Дочерние окна используют для создания элементов управления. Определяемые системой классы элементов управления представляют собой дочерние окна. Базовый стиль дочерних окон описан константой WS\_CHILD. Этот стиль полностью совпадает со стилем WS\_CHILDWINDOW. Дочерние окна не имеют кнопок минимизации и максимального увеличения размера, но всегда имеют окно-родителя. Они «прилипают» к поверхности родителя, перемещаются с ним и не могут выйти за пределы родительского окна.

# Описание программ

## Задание №1

Сначала мы создаем два LISTBOX’a и заполняем один из них.

Листинг 1. Создание LISTBOX’ов

|  |
| --- |
| case WM\_CREATE:  {  hListL = CreateWindow("listbox", NULL, WS\_CHILD | WS\_VISIBLE | LBS\_STANDARD, 30, 30, 200, 100, hWnd, (HMENU)ID\_LISTL, hInst, NULL);  SendMessage(hListL, WM\_SETREDRAW, FALSE, 0L);  SendMessage(hListL, LB\_ADDSTRING, 0, (LPARAM)(LPSTR)"Первый");  SendMessage(hListL, LB\_ADDSTRING, 0, (LPARAM)(LPSTR)"Второй");  SendMessage(hListL, LB\_ADDSTRING, 0, (LPARAM)(LPSTR)"Третий");  SendMessage(hListL, WM\_SETREDRAW, TRUE, 0L);  hListR = CreateWindow("listbox", NULL, WS\_CHILD | WS\_VISIBLE | LBS\_STANDARD, 350, 30, 200, 100, hWnd, (HMENU)ID\_LISTR, hInst, NULL);  SendMessage(hListR, WM\_SETREDRAW, TRUE, 0L);  HWND ToRight = CreateWindow("BUTTON", "Переместить»", WS\_CHILD | WS\_VISIBLE, 240, 40, 100, 30, hWnd, (HMENU)ID\_ToRight, hInst, NULL);  HWND ToLeft = CreateWindow("BUTTON", "«Переместить", WS\_CHILD | WS\_VISIBLE, 240, 80, 100, 30, hWnd, (HMENU)ID\_ToLeft, hInst, NULL);  HWND TextBtn = CreateWindow("BUTTON", "Настроить", WS\_CHILD | WS\_VISIBLE, 600, 30, 100, 30, hWnd, (HMENU)ID\_SETTINGS, hInst, NULL);  };break; |

Затем по нажатию кнопки переносим выделенный элемент LISTBOX’a в другой список.

Листинг 2. Основная логика задания

|  |
| --- |
| case ID\_ToRight:  {  // Определяем номер выделенной строки  if ((num = SendMessage(hListL, LB\_GETCURSEL, 0, 0L) )!= LB\_ERR)  {  // Получаем выделенную строку  SendMessage(hListL, LB\_GETTEXT, num, (LPARAM)tempText);  //Добавляем строку в правый список  SendMessage(hListR, LB\_ADDSTRING, 0, (LPARAM)(LPSTR)tempText);  //Удаляем строку из левого списка  SendMessage(hListL, LB\_DELETESTRING, num, (LPARAM)(LPSTR)tempText);  }  };break;  case ID\_ToLeft:  {  if ((num = SendMessage(hListR, LB\_GETCURSEL, 0, 0L)) != LB\_ERR)  {  SendMessage(hListR, LB\_GETTEXT, num, (LPARAM)tempText);  //Добавляем строку в левый список  SendMessage(hListL, LB\_ADDSTRING, 0, (LPARAM)(LPSTR)tempText);  //Удаляем строку из правого списка  SendMessage(hListR, LB\_DELETESTRING, num, (LPARAM)(LPSTR)tempText);  }  };break; |

## Задание №2

Для сохранения атрибутов текста создана специальная структура Text\_Set.

Листинг 3. Структура с атрибутами текста

|  |
| --- |
| struct Text\_Set {  TCHAR TEXT[1024];  COLORREF TextColor = NULL;  COLORREF BGColor = NULL;  int orientation = 0;  }; |

Сперва, с помощью конструктора добавляем диалоговое окно в проект, переносим на него нужные нам элементы и редактируем его внешний вид.

Затем в функции главного окна, нужно вызвать метод DialogBox для создания диалогового окна.

Листинг 4. Создание диалогового окна

|  |
| --- |
| case ID\_SETTINGS: {  int iCode = DialogBox(hInst, MAKEINTRESOURCE(DLG\_SET), hWnd, SetDLG);  if (iCode == IDOK)  {  InvalidateRect(hWnd, 0, TRUE);  UpdateWindow(hWnd);  }  };break; |

В функции диалогового окна перед его отображением для элементов устанавливаем значения, если они есть.

Листинг 5. Функция диалогового окна

|  |
| --- |
| INT\_PTR CALLBACK SetDLG(HWND hDlg, UINT msg, WPARAM wParam, LPARAM lParam)  {  int k=0;  HWND List = GetDlgItem(hDlg, IDC\_ORIENTATION);  switch (msg)  {    case WM\_INITDIALOG:  {  SendMessage(List, LB\_ADDSTRING, 0, (LPARAM)(LPSTR)"По вертикали");  SendMessage(List, LB\_ADDSTRING, 0, (LPARAM)(LPSTR)"По горизонтали");    SetDlgItemText(hDlg, IDC\_Text, str.TEXT);  \_itoa\_s(str.TextColor, tempText, 10);  SetDlgItemText(hDlg, IDC\_TextCode, tempText);  \_itoa\_s(str.BGColor, tempText, 10);  SetDlgItemText(hDlg, IDC\_BGCode, tempText);  if (str.orientation == 0)  {  SendMessage(List, LB\_SETCURSEL, 1, 0L);  }  else  {  SendMessage(List, LB\_SETCURSEL, 0, 0L);  }  return TRUE;  };  case WM\_COMMAND:  switch (LOWORD(wParam))  {    case IDOK:  {  GetDlgItemText(hDlg, IDC\_Text, str.TEXT, 1024);  GetDlgItemText(hDlg, IDC\_TextCode, tempText, 10);  str.TextColor = atoi(tempText);  GetDlgItemText(hDlg, IDC\_BGCode, tempText, 10);  str.BGColor = atoi(tempText);  if ((k = SendMessage(List, LB\_GETCURSEL, 0, 0L)) != LB\_ERR)  {  SendMessage(List, LB\_GETTEXT, k, (LPARAM)tempText);  }  if (!strcmp(tempText, "По горизонтали") )  str.orientation = 0;  else if(!strcmp(tempText, "По вертикали"))  str.orientation = 1;  EndDialog(hDlg, IDOK);  return TRUE;  };    case IDCANCEL: {  EndDialog(hDlg, IDCANCEL);  return TRUE;  }  case IDC\_TEXTCOLOR:  {  col(hDlg);  str.TextColor = colClndr;  \_itoa\_s(str.TextColor, tempText, 10);  SetDlgItemText(hDlg, IDC\_TextCode, tempText);  return TRUE;  };    case IDC\_BGCOLOR:  {  col(hDlg);  str.BGColor = colClndr;  \_itoa\_s(str.BGColor, tempText, 10);  SetDlgItemText(hDlg, IDC\_BGCode, tempText);  return TRUE;  };  }  }  return FALSE;  } |

Для выбора цвета текста и заднего фона вызывается окно палитры цветов.

Листинг 6. Создание окна палитры цветов

|  |
| --- |
| void col(HWND hwnd)  {  CHOOSECOLOR ch\_color = { 0 };  COLORREF cust\_colors[16] = { 0 };  ch\_color.Flags = CC\_FULLOPEN | CC\_RGBINIT ;  ch\_color.lStructSize = sizeof(CHOOSECOLOR);    ch\_color.hwndOwner = hwnd;    ch\_color.lpCustColors = cust\_colors;  if (ChooseColor(&ch\_color))  {  colClndr = ch\_color.rgbResult;  }    } |

Если диалоговое окно закрылось по кнопке OK, то все окно приложения перерисовывается. При обработке сообщения WM\_PAINT программа создает пользовательский шрифт и определяет выбор ориентации текста, а затем устанавливает угол наклона для выводимой строки.

Листинг 7. Обработка сообщения WM\_PAINT

|  |
| --- |
| case WM\_PAINT:  {  GetClientRect(hWnd, &rect);  HDC hDC;  PAINTSTRUCT ps;  HGDIOBJ hfnt, hfntPrev;  HRESULT hr;  hDC = BeginPaint(hWnd, &ps);  PLOGFONT plf = (PLOGFONT)LocalAlloc(LPTR, sizeof(LOGFONT));    if (str.orientation == 1)  {  plf->lfEscapement = 2700;  }  else  {  plf->lfEscapement = 0;  }  plf->lfWeight = 700;  strcpy(plf->lfFaceName, "Segoe UI");  hfnt = CreateFontIndirect(plf);  hfntPrev = SelectObject(hDC, hfnt);  SetTextColor(hDC, str.TextColor);  SetBkColor(hDC, str.BGColor);  TextOut(hDC, 600, 80, str.TEXT, strlen(str.TEXT));  SelectObject(hDC, hfntPrev);  DeleteObject(hfnt);  EndPaint(hWnd, &ps);  }; break; |

# Контрольные примеры работы программ

## Задание №1

Рис. . Окно приложения после запуска

На рисунке 2 показан результат перемещения элементов из одного списка в другой.

Рис. . Изменения в LISTBOX'ах

## Задание №2

На рисунках 3 и 4 отображены диалоговое окно и результат создания текста с выбранными атрибутами.

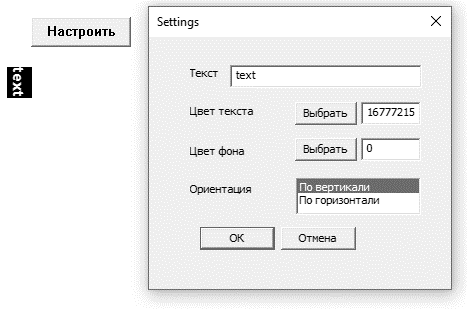
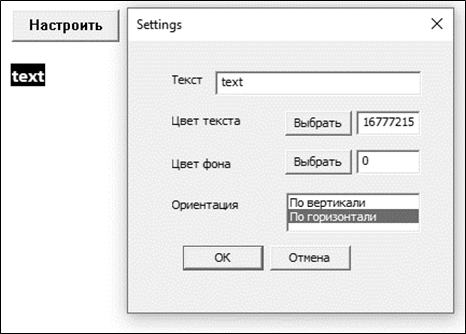
Рис. 4. Текст по горизонтали

Рис. . Текст по вертикали

# Список литературы

1. Белов. Е. А. Программирование приложений Windows. API графического интерфейса пользователя: учеб. пособие / Е. А. Белов. – Брянск: БГТУ, 2010. – 350 с.
2. Рихтер, Дж. Windows для профессионалов: создание эффективных Win32-приложений (с CD-ROM) / Дж. Рихтер. – М.: Русская редакция, 2000. – 752 с.
3. Рихтер, Дж. WINDOWS для профессионалов. Создание эффективных WIN32-приложений с учетом специфики 64-разрядной версии WINDOWS: [перевод] / Дж. Рихтер – 4-е изд. – СПб. [и др.]: Питер, Русская редакция, 2001. – 722 с.
4. Инструменты разработки и языки программирования [Электронный ресурс]. – Режим доступа: http://msdn.microsoft.com/. – Загл. с экрана. – Яз. рус.
5. RSDN [Электронный ресурс]. – Режим доступа: http://www.rsdn.ru/. – Загл. с экрана. – Яз. рус.