Федеральное агентство образования
Димитровградский институт технологий,
управления и дизайна

Лабораторная работа №1,2,3,5
по курсу: "Программирование для Windows NT
на языке 'C++' с использованием библиотеки MFC"

Выполнил студент гр.ВТ-31:
Потеренко А.Г.
Проверил преподаватель:
Наскальнюк А.Н.

Димитровград 2006г.

Содержание

Задание к лабораторным работам

Лабораторная работа №1
Создать программу, выводящую на экран надпись в рабочем окне программы "Hello Windows NT" разными цветами.
Цель работы:
Приобрести практические навыки создание простых приложений под Windows NT. Освоить работу основных блоков программы таких, как очередь сообщений, оконная процедура, регистрация класса окна, цикл обработки сообщений и обработка сообщений программы. Научиться выводить текст на экран рабочего окна программы, задавать различный фон окна, цвет надписи (выводимой информации в окне) "Hello Windows NT".

Лабораторная работа №2
Создать программу, выводящую на экран различными цветами и стилями элементы графики.
Цель работы:
Разобраться с работой с сообщением WM_PAINT, приобрести навыки работы с элементами графики.

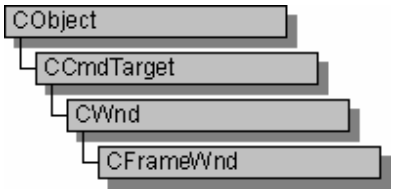
Лабораторная работа №3
Добавить в программу созданной на первой, второй лабораторной работе горизонтальные и вертикальные линии прокрутки.
Цель работы:
Разобраться с работой сообщений WM_VSCROLL, WM_HVSCROLL.

Лабораторная работа №5
Добавить в программу созданной лабораторной работе №4 окно диалога о программе (About). Окно диалога должно иметь две кнопки OK и Cancel, причем одна из кнопок должна быть нестандартной формы (эллипс, ромб). Для кнопки нестандартной формы реакция должна осуществляться только при попадании указателя мыши во внутреннюю часть.
Цель работы:
Изучить работу с диалогами и его элементами, курсорами, иконками, способы создания нестандартных объектов.

Алгоритм решения и теория к программам

Теория к ЛР №1

CFrameWnd



CFrameWnd – класс обеспечивает функциональные возможности окон интерфейсом SDI. Чтобы создавать полезное окно с рамкой для вашего приложения, нужно получить класс, производный из CFrameWnd. Добавьте переменные члены к полученному классу, чтобы хранить данные, определенные в вашем приложении. Определите функции члены сообщений и карту сообщения в полученном классе, чтобы определить то, что случается, когда сообщения идут к окну.

Подключаемый заголовочный файл:

#include "afxwin.h"

Наша программа создает класс CFrameWnd_LAB1, производный от данного. В конструкторе:

CFrameWnd_LAB1()

создается окно. Затем загружается и устанавливается меню из ресурсов. В методе:

void OnPaint()

создается объект dc(this) класса CPaintDC, рисуется прямоугольник в окне фона COLOR_3DDKSHADOW. Метод GetClientRect определяет клиентскую область окна. И если пользователь захотел вывести текст, то вызывается метод

LAB1_MENU_P1(dc,200,100,50)

Метод

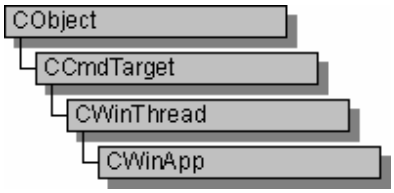
void LAB1_MENU_P1(HDC hdc,short int R, short int G,short int B)

выводит сообщение в окно программы.

Карта сообщений для CFrameWnd_LAB1:

```
BEGIN_MESSAGE_MAP(CFrameWnd_LAB1, CFrameWnd)
      ON_WM_PAINT()
      ON_COMMAND(ID_MENUITEM40001,LAB1_MENU_FUNC2)
      ON_COMMAND(ID_MENUITEM40002,LAB1_MENU_FUNC1)
      ON_COMMAND(ID_MENUITEM40003,LAB1_MENU_FUNC3)
END_MESSAGE_MAP()
```

CWinApp



CWinApp класс – класс из которого вы получаете объект приложения Windows. Объект обеспечивает функцией

virtual BOOL InitInstance()

для инициализации вашего приложения.

В нашей программе определяется класс **CWinApp_LAB1**, производный от данного. В нем перегружается метод

virtual BOOL InitInstance()

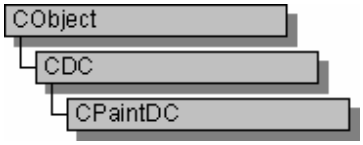
```
{
        m_pMainWnd=new CFrameWnd_LAB1();
        ASSERT(m_pMainWnd);
        m_pMainWnd->ShowWindow(SW_SHOW);
        m_pMainWnd->UpdateWindow();
        return TRUE;
}
```

Затем создаем глобальный объект

```
CWinApp_LAB1 theApp
```

Теория к ЛР №2

CPaintDC



CPaintDC класс – класс контекста устройства, полученный из CDC. Он выполняет CWnd:: BeginPaint во время начала работы конструктора и CWnd::EndPaint во время работы деструктора. Объект CPaintDC может только использоваться при ответе на WM_PAINT сообщение, обычно в вашей OnPaint функции сообщения.

Наша программа создает класс **CPaintDC_LAB2** с ключом **public**, производный от данного.

Метод

DRAW_LAB2()

этого класса рисует на поверхности окна фигуры, которые необходимо нарисовать по заданию.

Метод

void CFrameWnd_LAB2::OnPaint()

создает объект

CPaintDC_LAB2 dc(this)

Подключаемый заголовочный файл:

#include "afxwin.h"

Карта сообщений для CFrameWnd_LAB2:

```
BEGIN_MESSAGE_MAP(CFrameWnd_LAB2, CFrameWnd)
    ON_WM_PAINT()
    ON_COMMAND(ID_MENUITEM40001,LAB2_MENU_FUNC2)
    ON_COMMAND(ID_MENUITEM40002,LAB2_MENU_FUNC1)
    ON_COMMAND(ID_MENUITEM40003,LAB2_MENU_FUNC3)
END_MESSAGE_MAP()
```

Теория к ЛР №3

В данной работе необходимо добавить полосы прокрутки. Для этого необходимо создать окно с соответствующим стилем. Необходимо добавить в карту сообщений соответствующие записи:

```
BEGIN_MESSAGE_MAP(CFrameWnd_LAB3, CFrameWnd)
    ON_WM_PAINT()
    ON_COMMAND(ID_MENUITEM40001,LAB3_MENU_FUNC2)
    ON_COMMAND(ID_MENUITEM40002,LAB3_MENU_FUNC1)
    ON_COMMAND(ID_MENUITEM40003,LAB3_MENU_FUNC3)
    ON_WM_VSCROLL()
    ON_WM_HSCROLL()
    ON_WM_CREATE()
    ON_WM_SIZE()
END_MESSAGE_MAP()
```

Для понимания работы программы нужно определить некоторые понятия. Пусть у нас виртуальное окно – то, которое не помещается в клиентскую область и его нужно все просмотреть. Пусть также имеется глобальная точка начала

(x_BUF,y_BUF)

относительно которой выводится все, что должно выводиться в рабочую область. То есть при выводе изображений, их координаты переносятся на вектор

(–x_BUF,–y_BUF)

Определим

MAX_PLOSHAD_B 1000

как максимальную высоту виртуального окна, а

MAX_PLOSHAD_A 200

как максимальную длину виртуального окна.
Метод

int OnCreate(LPCREATESTRUCT lpCreateStruct)

устанавливает диапазон прокрутки и текущее положение ползунка.
При увеличении или уменьшении размеров окна диапазон прокрутки должен меняться:

```
b=ceil((768/cy)*MAX_PLOSHAD_B);  //768 – разрешение по вертикали окна
a=ceil((1024/cy)*MAX_PLOSHAD_A); //1024 – разрешение по горизонтали
```

Ну и соответственно методы

void OnVScroll(UINT nSBCode, UINT nPos, CScrollBar* pScrollBar)

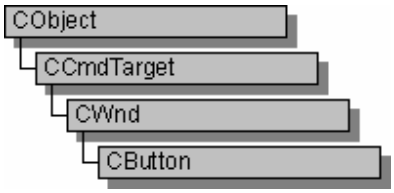
и

void OnHScroll(UINT nSBCode, UINT nPos, CScrollBar* pScrollBar)

меняют положение ползунка и координаты глобальной точки в зависимости от действий пользователя.

Теория к ЛР №5

CButton



CButton – класс обеспечивает функциональные возможности управления кнопкой Windows. Кнопка – маленькое, прямоугольное дочернее окно, на которую можно нажать. Кнопки могут использоваться по одной или в группах и могут появляться без текста. Кнопка реагирует на события, когда пользователь нажимает по ним.

Подключаемый заголовочный файл:

#include "afxwin.h"

Каждое событие карты сообщения имеет следующую форму:

ON_Notification(*id*, *memberFxn*)

где *id* определяет дочернее окно ID управления, посылающего уведомление, и *memberFxn* – имя родительской функции члена, которую вы написали, чтобы обратиться с уведомлением.

Образец функции родителя следующий:

afx_msg void *memberFxn***( );**

Потенциальные сообщения карты сообщений:

Карта сообщений	**Посылается родителю, когда...**
ON_BN_CLICKED	Пользователь нажимает кнопку
ON_BN_DOUBLECLICKED	Пользователь дважды нажимает кнопку

В нашей программе определяется класс **CBUTTONS**, производный от данного. Метод

void REG()

создает регион для кнопки. У нас задача создать овальную кнопку, поэтому создаем регион овальной формы. Теперь кнопка реагирует на события только во внутренней части овала кнопки. Метод

void BUTTON_DRAW(bool FLAG)

рисует саму кнопку и все изменения, происходящие с ней.

CDialog

CDialog класс – класс, используемый для показа окно диалога на экране. Диалоги имеют два типа: модальный и не зависящий от режима. Модальный диалог должен быть закрыт пользователем прежде, чем приложение продолжит работу. Диалог, не зависящий от режима, позволяет пользователю показывать диалог и возвращать управление к другой задаче без того, чтобы отменить или закрыть диалог.

Подключаемый заголовочный файл:

#include "afxwin.h"

Наша программа создает класс **CDialog_LAB5**, производный от данного. В данном классе определяется

CBUTTONS * ButtonS

Метод

```
afx_msg void OnDrawItem(int nIDCtl, LPDRAWITEMSTRUCT lb)
```

отвечает на событие при перерисовки кнопки ButtonS. Метод

```
void EXIT()
```

отвечает на нажатие овальной кнопки.
В методе

```
void CDialog_LAB5::DoDataExchange(CDataExchange* pDX)
```

создается кнопка ButtonS:

```
ButtonS->Create("",
                WS_CHILD | WS_VISIBLE | BS_OWNERDRAW,
                CRect(325,55,455,105),this,BUTO1);
```

и вызывается метод

```
ButtonS->REG()
```

Карта сообщений для CDialog_LAB5:

```
BEGIN_MESSAGE_MAP(CDialog_LAB5, CDialog)
      //{{AFX_MSG_MAP(CDialog_LAB1)
      //}}AFX_MSG_MAP
      ON_WM_DRAWITEM()
      ON_BN_CLICKED(BUTO1,EXIT)
END_MESSAGE_MAP()
```