Windows programming

1 Инструменты

1. Borland Pascal for Windows.
2. Под си любой удобный редактор.

2 Пример простейшего Windows приложения

Практически каждая Windows программа состоит из ряда обязательных частей:

1. Функция WinMain — точка входа в программу.
2. Создание и описание атрибутов класса окна
3. Создание экземпляра окна данного класса
4. Цикл обработки сообщений
5. Оконная функция, обрабатывающая сообщения

Рассмотрим пример программы:

```
uses
    WinTypes, WinProcs;
var
    Wnd: HWND;
    Msg: TMsg;
function Window1(Wnd: HWND;
                 iMessage, wParam: Word;
                 LParam:LongInt):LongInt; export;
begin
    case iMessage of
    WM_DESTROY:
        begin
            PostQuitMessage(0);
            exit;
        end;
    end;
    Window1:=DefWindowProc(Wnd, iMessage, wParam, LParam);
end;
```

Оконная функция выполняет непосредственную обработку сообщений. Параметры передаваемые этой функции эквивалентны полям структуры типа TMsg. Каждое получаемое оконной функцией сообщение имеет уникальный идентификатор. Для обработки сообщений используется конструкция типа переключателя. Все сообщения не обрабатываемые оконной функцией передаются специальной функции ядра ОС DefWindowProc. Описанное сообщение WM_DESTROY означает, что система пытается закрыть окно. В ответ на это сообщение вызовом процедуры PostQuitMessage программа помещает в очередь сообщение WM_QUIT и когда цикл обработки получает это сообщение, программа завершается.

```
procedure Register;
var
        w:TWndClass;
begin
        if (hPrevInst <> 0) then
                Exit;
        w.Style:=cs_HRedraw or cs_VRedraw;
        w.lpfnWndProc:=@Window1;
        w.cbClsExtra:=0;
        w.cbWndExtra:=0;
        w.hInstance:=hInstance;
        w.hCursor:=LoadCursor(0, idc_arrow);
        w.hIcon:=LoadIcon(0, idi_application);
        w.hbrBackground:=GetStockObject(white_brush);
        w.lpszMenuName:='';
        w.lpszClassName:='Class1';
        if not RegisterClass(w) then
        begin
                MessageBox(GetFocus,
                           'Cannot register class',
                           'Error', MB_OK);
                Halt;
        end;
end;

procedure WinMain;
begin
        Register;
        Wnd:=CreateWindow('Class1', 'Main Window',
                          WS_OverlappedWindow, 0, 0,
                          200, 400, 0, 0,
                          hPrevInst, nil);
        ShowWindow(Wnd, CmdShow);
        UpdateWindow(Wnd);
        while GetMessage(Msg, 0, 0, 0) do
        begin
                TranslateMessage(Msg);
                DispatchMessage(Msg);
        end;
end;
```

После того как окно отображено на экране, управление передается циклу обработки сообщений. Функция GetMessage извлекает сообщение из очереди и помещает в структуру типа TMsg. Для всех сообщений, кроме сообщения WM_QUIT, функция GEtMessage возвращает ненулевое значение и цикл продолжает обработку сообщений. Процедура TranslateMessage передает структуру типа TMsg ядру Windows для преобразования сообщений виртуальных клавиш в сообщения о символах. Процедура DispatchMessage используется для распределения текущего сообщения соответствующей функции окна.

```
Begin
        WinMain;
End.
```

3 Графический интерфейс

3.1 Интерфейс графических устройств

Интерфейс графических устройств это модуль программы, который обрабатывает операторы работы с графикой. По умолчанию он работает в режиме пиксельных координат. Перед началом работы необхрдимо получить контекст устройства — специальную структуру даннных, соде*ржащую остовные характеристики устройства, а так же различные средства отображения. Существуют 2 способа получения контеста устройства:*

1. Внутри сообщения WM_PAINT

```
HDC hwnd;
PAINTSTRUCT ps;
hdc = BeginPaint(hwnd,&ps);
//...// Происходит рисование
EndPaint(hwnd);//Освобождение контекста устройства посленрисования
```

2. Используется в тех случаях когда приходится рисовать в сообщения отличных от WM_PAINT

```
hdc = GetDC(hwnd);
//....// Происходит рисование
ReleaseDC(hwnd,hdc);
```

3.2 Инструменты рисования

3.2.1 Карандаш

```
HPEN pen = CreatePen(стиль, толщина, цвет);
SelectObject(hdc, pen);
```

Стиль

1. PS_SOLID
2. PS_DASH
3. PS_DOT
4. PS_DASHDOT
5. PS_DASHDOTDOT

Цвет

```
RGB(R, G, B);
```

Ширина Работает только для PS_SOLID. (Для всех остальных стилей ширина должна быть равна еденице)

3.2.2 Кисть

```
HBRUSH brush;
brush = CreateSolidBrush(цвет);
brush = CreateHatchBrush(стиль, цвет);

HBITMAP hBitmap;//Рисунок BMP
brush = CreatePatternBrush(hBitmap);
```

Стиль

1. HS_VERTICA
2. HS_HORISONTAL
3. HS_CROSS
4. HS_BDIAGONAL
5. HS_FDIAGONAL
6. HS_DIAGCROSS

3.3 Графические примитивы

Необходимо получить контекст устройства.

1. Rectangle(hdc, x1, y1, x2, y2);
2. Ellipse(hdc, x1, y1, x2, y2);
3. Arc(hdc, x1, y1, x2, y2, x3, y3, x4, y4);
   x3, y3, x4, y4 Задают угол.
4. Chord(hdc, x1, y1, x2, y2, x3, y3, x4, y4);
5. Pie(hdc, x1, y1, x2, y2, x3, y3, x4, y4);
6. RoundRect(hdc, x1, y1, x2, y2, x3, y3);
   x3, y3 — радиус закругления
7. Polyline(hdc,p,n); //Ломаная
8. Polygone(hdc,p,n); //Замктурый многоугольник

3.4 Сообщение от мыши.

WM_MOUSEMOVE
WM_LBUTTONDOWB
WM_RBUTTONDOWN
WM_LBUTTONUP
WM_RBUTTONUP

Для того чтобы ваще предложение обрабатывало 2йной клик необходимо добывать следующий стиль: CS_DBLCLKS

Координаты мышки ипередаются в параметре LParam.

Параметр WParam содержит слово, описывающее состояние клавиатуры

Содержание