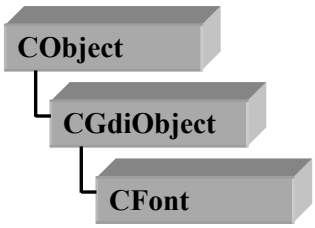
Л Е К Ц И Я 1 1

ШРИФТ. КЛАСС CFONT 1
Атрибуты шрифта. Структура LOGFON 1
Инициализация шрифта 3
Получение текущего шрифта. Функция GetCurrentFont 4
Получение атрибутов текущего шрифта. Функция GetLogFont 4
Пример получения атрибутов текущего шрифта 4
Завершение работы со шрифтом 4
Пример создания шрифта 4
ДИАЛОГОВОЕ ОКНО ВЫБОРА ШРИФТА 5
Конструктор класса CFontDialog 5
Отображение диалогового окна. Функция DoModal 6
Получение шрифта, выбранного пользователем 6
Пример получения шрифта, выбранного пользователем 6
Получение атрибутов шрифта, выбранного пользователем 6
Пример получения атрибутов шрифта, выбранного пользователем 7

ШРИФТ. КЛАСС CFONT

Класс **CFont** инкапсулирует графический объект Windows – “шрифт”. С помощью этого объекта можно изменять шрифт текста, отображаемого в элементе управления или нарисованного в форме. Иерархия классов относительно класса **CFont** представлена на рисунке.



Атрибуты шрифта. Структура LOGFON

Для хранения атрибутов шрифта в библиотеке MFC определена структура **LOGFON**. Структура содержит имя установленного в операционной системе шрифта и основные характеристики его начертания. Кроме того, на тот случай если в системе нет шрифта с заданным именем, структура содержит критерии поиска шрифта среди установленных шрифтов. Структура **LOGFON** объявлена следующим образом:

```
typedef struct tagLOGFONT
{
    LONG lfHieght ;              // высота символов или знакомест
    LONG lfWidth ;               // ширина символов
    LONG lfEscapement ;          // угол наклона символов
    LONG lfOrientation ;         // угол наклона строки
    LONG lfWeight ;              // толщина символов
    BYTE lfItalic ;              // курсивное начертание символов
    BYTE lfUnderline ;           // подчеркнутое начертание символов
    BYTE lfStrikeOut ;           // перечеркнутое начертание символов
    BYTE lfCharSet ;             // таблица кодировки символов
    BYTE lfOutPrecision ;        // критерий выбора шрифта
    BYTE lfClipPrecision ;       // критерий отсечения символов
    BYTE lfQulity ;              // качество вывода символов
    BYTE lfPitchAndFamily ;      // тип и семейство шрифтов
    TCHAR lfFaceName [32] ;      // имя шрифта
} LOGFONT , *PLOGFONT;
```

Члены структуры:

lfHieght определяет высоту символов или знакомест в логических единицах
= 0 для нахождения шрифта используется значение высоты по умолчанию

> 0 значение преобразуется в единицы устройства вывода и используется для выбора шрифта из имеющихся шрифтов, соответствующего по высоте знакоместа

< 0 значение преобразуется в единицы устройства вывода и используется для выбора шрифта из имеющихся шрифтов, соответствующего по высоте символа

lfWidth определяет среднюю ширину символов в логических единицах. Нулевое значение обеспечивает выбор шрифта из имеющихся шрифтов, наиболее подходящего к конкретному устройству по возможности масштабирования.

lfEscapement определяет угол в десятых долях градуса между базовой линией выводимого текста и горизонталью

lfOrientation определяет угол в десятых долях градуса между базовой линией каждого выводимого символа и горизонталью

lfWeight определяет толщину символов и может принимать значение по умолчанию. Некоторым значениям даны символьные наименования:

FW_DONTCARE	**0**	**FW_THIN**	**100**	**FW_EXTRALIGHT**	**200**
FW_ULTRALIGHT	**200**	**FW_LIGHT**	**300**	**FW_NORMAL**	**400**
FW_REGULAR	**400**	**FW_MEDIUM**	**500**	**FW_SEMIBOLD**	**600**
FW_DEMIBOLD	**600**	**FW_BOLD**	**700**	**FW_EXTRABOLD**	**800**
FW_ULTRABOLD	**800**	**FW_HEAVY**	**900**	**FW_BLACK**	**900**

lfItalic определяет курсивное начертание символов, если установлено значение **TRUE**

lfUnderline определяет подчеркнутое начертание символов, если установлено значение **TRUE**

lfStrikeOut определяет перечеркнутое начертание шрифта, если установлено значение **TRUE**

lfCharSet определяет таблицу кодировки шрифта. Может принимать одно из следующих значений:

ANSI_CHARSET	**BALTIC_CHARSET**	**CHINESEBIG5_CHARSET**
DEFAULT_CHARSET	**EASTEUROPE_CHARSET**	**GB2312_CHARSET**
GREEK_CHARSET	**HANGUL_CHARSET**	**MAC_CHARSET**
OEM_CHARSET	**RUSSIAN_CHARSET**	**SHIFTJIS_CHARSET**
SYMBOL_CHARSET	**TURKISH_CHARSET**	

lfOutPrecision определяет механизм выбора шрифта, т.е. насколько шрифт должен соответствовать заданным параметрам, прежде всего типу шрифта, а также ширине, высоте, ориентации и т.д. Это поле может принимать одно из следующих значений:

OUT_DEFAULT_PRECIS определяет механизм выбора, принятый по умолчанию;

OUT_DEVICE_PRECIS если в системе установлены несколько шрифтов с одним и тем же названием, должен быть выбран системный шрифт;

OUT_OUTLINE_PRECIS только для Windows NT: выбор осуществляется из набора векторных или шрифтов **TrueType**;

OUT_RASTER_PRECIS если в системе установлено несколько шрифтов с одним и тем же названием, должен быть выбран растровый шрифт;

OUT_STRING_PRECIS используется для получения списка векторных и шрифтов **TrueType**;

OUT_STROKE_PRECIS используется для получения списка векторных и шрифтов **TrueType**;

OUT_TT_ONLY_PRECIS задает выбор только шрифта **TrueType**, и если ни один такой шрифт не установлен, используется механизм выбора, принятый по умолчанию;

OUT_TT_PRECIS если в системе установлено несколько шрифтов с одним и тем же названием, должен быть выбран шрифт **TrueType**.

lfClipPrecision определяет механизм отсечения символов шрифта в случае, если символы частично оказываются вне области вывода. Это поле может принимать одно или несколько из следующих значений:

CLIP_DEFAULT_PRECIS используется механизм по умолчанию;

CLIP_STROKE_PRECIS используется только для получения списка растровых, векторных или шрифтов **TrueType**;

CLIP_EMBEDDED необходимо указывать этот флаг при использовании внедренных шрифтов;

CLIP_LH_ANGLES при использовании этого флага отсчет направления угла поворота (по часовой стрелке или против) зависит от выбранной системы координат; если флаг не используется, то поворот текста, выводимого с ис-

пользованием системных шрифтов, производится против часовой стрелки, однако для всех остальных шрифтов направление поворота определяется системой координат;

lfQulity определяет качество вывода, которое зависит от выбранного шрифта. Оно может принимать следующие значения:

DEFAULT_QUALITY	качество вывода текста не критично;
DRAFT_QUALITY	качество вывода текста менее важно, чем соответствие заданным атрибутам;
PROOF_QUALITY	качество вывода текста более важно, чем соответствие заданным атрибутам.

lfPitchAndFamily определяет тип и семейство шрифтов. Для задания типа и семейства используется операция OR. Тип шрифта определяется двумя младшими битами поля и может принимать одно из следующих значений:

DEFAULT_PITCH	тип, принятый по умолчанию;
VARIABLE_PITCH	пропорциональный шрифт.
FIXED_PITCH	непропорциональный шрифт (все символы имеют одинаковый размер);

Семейство определяется старшими четырьмя битами поля и может принимать одно из следующих значений

FF_DECORATIVE	декоративный шрифт;
FF_DONTCARE	семейство не имеет значения;
FF_MODERN	непропорциональный шрифт с засечками или без них;
FF_ROMAN	пропорциональный шрифт с засечками;
FF_SCRIP	шрифт, схожий по начертанию с рукописным;
FF_SWISS	пропорциональный шрифт без засечек.

lfFaceName имя шрифта; объект типа **CString** или указатель на строку, заканчивающуюся нуль-терминатор (не более 32 символов, включая нуль-терминатор).

Инициализация шрифта

Для создания объекта этого класса имеется конструктор без параметров, потому для его инициализации необходимо вызвать одну из следующих функций: **CreateFont**, **CreateFontIndirect**, **CreatePointFont**, **CreatePointFontIndirect**. Для использования созданного шрифта надо установить его в конкретный контекст устройства.

Прототип функции **CreateFont**, параметры которой принимают те же значения, что и поля структуры **LOGFONT**, имеет вид:

```
BOOL CreateFont ( int nHeight , int nWidth , int nEscapement , int nOrientation , int nWeight ,
                  BYTE bItalic , BYTE bUnderline , BYTE cStrikeOut , BYTE nCharSet ,
                  BYTE nOutPrecision , BYTE nClipPrecision , BYTE nQuality ,
                  BYTE nPitchAndFamily , LPCTSTR lpszFacename ) ;
```

Следующая функция создает шрифт на основе структуры **LOGFONT**. Прототип функции имеет вид:

```
BOOL CreateFontIndirect ( const LOGFONT* lpLogFont ) ;
```

В качестве параметра используется указатель на структуру **LOGFONT**, которая определяет атрибуты логического шрифта.

Следующая функция создает шрифт на основе только его имени и устанавливает размер шрифта:

```
BOOL CreatePointFont ( int nPointSize , LPCTSTR lpszFaceName , CDC* pDC = NULL ) ;
```

Параметры:

nPointSize определяет величину шрифта в десятых долях типографского пункта

lpszFaceName имя шрифта, объект типа **CString** или указатель на строку, заканчивающуюся нуль-терминатором (не более 30 символов). Если параметр недействителен, то используется шрифт, независимый от устройства

pDC указатель на объект типа **CDC**, который используется, чтобы преобразовать высоту символов **nPointSize** в логические единицы. Если параметр недействителен, то используется контекст устройства экрана

Следующая функция подобна функции **CreateFontIndirect** за исключением того, что член **lfHeight** структуры **LOGFONT** вместо логических единиц устройства определяет высоту символа в десятых долях типографского пункта. Прототип функции имеет вид:

BOOL CreatePointFontIndirect (const LOGFONT* lpLogFont , CDC* pDC = NULL) ;

Получение текущего шрифта. Функция GetCurrentFont

Для получения текущего шрифта в классе **CDC** определена функция **GetCurrentFont**, прототип которой имеет вид:

CFont* GetCurrentFont ( ) const ;

Функция возвращает указатель в текущий шрифт.

Получение атрибутов текущего шрифта. Функция GetLogFont

Для получения атрибутов текущего шрифта в классе **CFont** определена функция **GetLogFont**, прототип которой имеет вид:

int GetLogFont (LOGFONT * pLogFont) ;

Параметры:

pLogFont указатель на структура **LOGFONT**, в которую копируются атрибуты шрифта.

Пример получения атрибутов текущего шрифта

```
// объявление экземпляра класса CClientDC для работы с клиентской областью текущего окна
CClientDC dc ( this ) ;
CFont* pF ;                                  // объявляем указатель на текущий фонт
pF = dc.GetCurrentFont () ;                  // инициализируем указатель на текущий фонт
if ( pF )                                    // если указатель определён
{
    LOGFONT lf ;                             // объявляем структуру, содержащую атрибуты фонта
    pF->GetLogFont ( &lf ) ;                 // получаем атрибуты фонта
    CString str ;                            // объявляем строку сообщения
    str.Format ("Имя шрифта - %s" , lf.lfFaceName ) ;   // формируем строку сообщения
    MessageBox ( str ) ;                     // выводим на экран окно сообщения
}
```

Завершение работы со шрифтом

После завершения работы со шрифтом его нужно заменить ранее установленным шрифтом. Удалить фонт из оперативной памяти можно с помощью функции **DeleteObject**, определённой в классе **CGdiObject** и у всех дочерних классов соответственно. Прототип функции **DeleteObject** имеет вид:

BOOL DeleteObject ( ) ;

Объекты класса **CGdiObject** или его потомков должны удаляться независимо друг от друга. Например, если удалена кисть, то точечный рисунок, связанный с кистью, не удаляется автоматически. Он должен быть удален независимо.

Пример создания шрифта

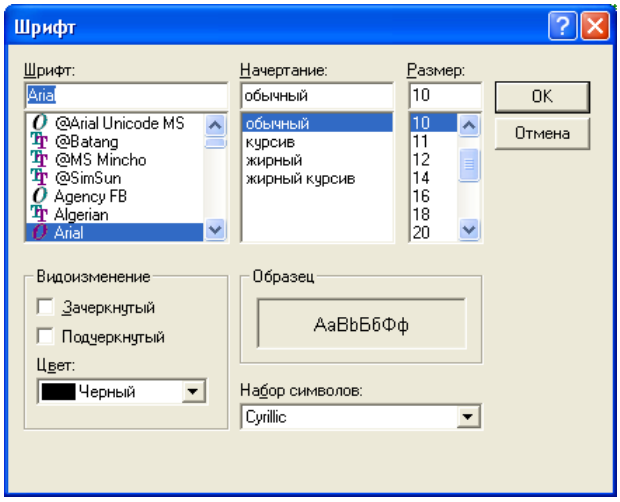
```
CFont font ;                    // объявляем объект шрифт
font.CreateFont (               // создаём шрифт с указанными атрибутами
    25 ,                        // nHeight              высота символов
    20 ,                        // nWidth               ширина символов
    50 ,                        // nEscapement          угол наклона строки - 5°
    0 ,                         // nOrientation
    FW_BOLD ,                   // nWeight              толщина символов - жирные
    TRUE ,                      // bItalic              курсив
    FALSE ,                     // bUnderline           не подчеркнутые
    FALSE ,                     // cStrikeOut           не перечеркнутые
    ANSI_CHARSET ,              // nCharSet             таблица кодировки
    OUT_DEFAULT_PRECIS ,        // nOutPrecision        критерий выбора шрифта
    CLIP_DEFAULT_PRECIS ,       // nClipPrecision       критерий отсечения символов
    DEFAULT_QUALITY ,           // nQuality             качество вывода символов
```

```
    DEFAULT_PITCH ,          // nPitchAndFamily      семейство шрифтов
    "Times New Roman" ) ;    // lpszFacename         имя шрифта
// объявляем экземпляр класса CClientDC для работы с клиентской областью текущего окна
CClientDC dc ( this ) ;
CFont* def_font ;            // объявляем указатель на ранее установленный шрифт
// выбираем созданный нами шрифт для использования и запоминаем  ранее установленный шрифт
def_font = dc.SelectObject ( &font ) ;
dc.TextOut ( 25 , 25 ,"Hello") ;   // рисуем строку в форме
dc.SelectObject ( def_font ) ;     // выбираем для использования ранее установленный шрифт
font.DeleteObject () ;             // удаляем созданный нами шрифт
```

В результате работы приведённого фрагмента кода в форме будет нарисована следующая строка:

ДИАЛОГОВОЕ ОКНО ВЫБОРА ШРИФТА

Стандартное диалоговое окно **Выбора шрифта** позволяет пользователю выбрать шрифт, его размер, стиль, цвета и предоставляет средства для получения информации о сделанном выборе. Диалоговое окно имеет вид, показанный на рисунке.



Конструктор класса CFontDialog

Конструктор класса имеет следующий прототип:

CFontDialog (LPLOGFONT lplfInitial = NULL , DWORD dwFlags = CF_EFFECTS | CF_SCREENFONTS , CDC* pdcPrinter = NULL , CWnd* pParentWnd = NULL) ;

Параметры:

lplfInitial — указатель на структуру типа **LOGFONT**, которая позволяет предварительно задать некоторые характеристики шрифта;

dwFlags — комбинация флагов, управляющих видом диалогового окна; некоторые флаги приведены ниже:

- **CF_EFFECTS** — позволяет устанавливать эффекты начертания: подчеркивание, перечеркивание и цвет;
- **CF_SCKEENFONTS** — отображается только список оконных шрифтов, поддерживаемых операционной системой;
- **CF_SHOWHELP** — диалоговое окно содержит кнопку **Help**;
- **CF_TTONLY** — диалоговое окно содержит список шрифтов только типа **True Type**;

CF_FORCEFONTEXIST диалоговое окно выводит сообщение об ошибке, если потребитель пытается выбрать шрифт или стиль, который не существует;

pdcPrinter указатель на контекст принтера;

pParentWnd указатель на родительское окно.

Отображение диалогового окна. Функция DoModal

Функция **DoModal**, определённая в классе **CFontDialog**, создает и отображает диалоговое окно выбора шрифта.

Синтаксис:

```
virtual int DoModal ( ) ;
```

Возвращаемое значение: IDOK или **IDCANCEL** – это константы, которые указывают, выбрал ли пользователь кнопку **OK** или **Отмена**.

Получение шрифта, выбранного пользователем

Все атрибуты выбранного шрифта содержатся в структуре **LOGFONT**. Получить эту структуру позволяет функция **GetCurrentFont**, определенных в классе **CFontDialog** :

```
void GetCurrentFont ( LPLOGFONT lplf ) ;
```

Параметры:

lplf указатель на структуру типа **LOGFONT**

Функция копирует атрибуты выбранного шрифта в переменную, на которую ссылается указатель **lplf**.

Пример получения шрифта, выбранного пользователем

Приведённый ниже фрагмент кода выводит на экран диалоговое окно выбора шрифта, в котором пользователь может указать атрибуты нового шрифта. Если пользователь закрывает окно с помощью кнопки **OK**, то в окне сообщений выводится название выбранного шрифта, которое содержится в поле **lfFaceName** структуры **lf**.

```
// объявляем экземпляр класса диалогового окна выбора шрифта
CFontDialog dlg ;
// выводим на экран диалоговое окно выбора шрифта
if ( dlg.DoModal () == IDOK )
{
        LOGFONT lf ;                          // объявляем шрифт
        dlg.GetCurrentFont ( &lf ) ;          // получаем выбранный шрифт
                                              // выводим на экран окно сообщений
        MessageBox ( "Face name of the selected font : " + lf.lfFaceName ) ;
}
```

Получение атрибутов шрифта, выбранного пользователем

Графический объект шрифт **CFont** и связанная с ним структура **LOGFONT** содержат достаточно много полей. Поэтому некоторых случаях удобно оперировать не со всем объектом, а только с определёнными его свойствами. Получить некоторые свойства выбранного шрифта можно с помощью следующих функций, определенных в классе **CFontDialog** :

```
CString GetFaceName ( ) const ;        // возвращает имя выбранного шрифта
CString GetStyleName ( ) const ;       // возвращает имя стиля выбранного шрифта
int GetSize ( ) const ;                // возвращает размер выбранного шрифта
COLORREF GetColor ( ) const ;          // возвращает цвет выбранного шрифта
int GetWeight ( ) const ;              // возвращает толщину выбранного шрифта
BOOL IsStrikeOut ( ) const ;           // возвращает 1, если выбран перечёркнутый шрифт
BOOL IsUnderline ( ) const ;           // возвращает 1, если выбран подчёркнутый шрифт
BOOL IsBold ( ) const ;                // возвращает 1, если выбран полужирный шрифт
BOOL IsItalic ( ) const ;              // возвращает 1, если выбран курсивный шрифт
```

Пример получения атрибутов шрифта, выбранного пользователем

Приведённый ниже фрагмент кода выводит на экран диалоговое окно выбора шрифта, в котором пользователь может указать атрибуты нового шрифта. Если пользователь закрывает окно с помощью кнопки **OK**, то формируются подстроки строки сообщения, каждая из которых содержит имя и значение атрибута выбранного шрифта. Затем эта строка выводится в окне сообщений.

```
// объявляем подстроки строки сообщения
CString size , name , color , weight , strike , under , bold , italic ;
// объявляем экземпляр класса диалогового окна выбора шрифта
CFontDialog dlg ;
// выводим на экран диалоговое окно выбора шрифта
if ( dlg.DoModal () == IDOK )        // если при закрытии окна пользователь нажал кнопку OK
{
// формируем подстроки строки сообщения
    name="Font\t\t" + dlg.GetFaceName() + "\n" ;                    // имя
    size.Format ("Size\t\t%i\n" , dlg.GetSize() / 10 ) ;            // размер
    color.Format ("Color\t\t0x%06X\n" , dlg.GetColor() ) ;          // цвет
    weight.Format ("Weight\t\t%i\n" , dlg.GetWeight () ) ;          // толщина
    strike.Format ("StrikeOut\t\t%i\n" , dlg.IsStrikeOut () ) ;     // перечёркнутый
    under.Format ("Underline\t\t%i\n" , dlg.IsUnderline () ) ;      // подчёркнутый
    bold.Format ("Bold\t\t%i\n" , dlg.IsBold () ) ;                 // жирный
    italic.Format ("Italic\t\t%i\n" , dlg.IsItalic () ) ;           // курсив
// выводим на экран окно сообщений
    MessageBox ( name + size + color + weight + strike + under + bold ) ;
}
```

В результате работы этого фрагмента кода на экран будет выведено следующее окно сообщений.