ВЛАДИМИР ДАВЫДОВ

Visual C++

Разработка Windows-приложений с помощью MFC и API-функций

Низкоуровневая и высокоуровневая технологии программирования

Редактор ресурсов. Элементы интерфейса

Динамически подключаемые библиотеки

Мастера. Каркасы Windows-приложений

Более 35 демонстрационных примеров, 280 вопросов и упражнений

+ CD

Владимир Давыдов

Visual C++

Разработка Windows-приложений с помощью MFC и API-функций

Санкт-Петербург
«БХВ-Петербург»
2008

УДК 681.3.068
ББК 32.973.26-018.1
Д13

Давыдов В. Г.

Д13 Visual C++. Разработка Windows-приложений с помощью MFC и API-функций. — СПб.: БХВ-Петербург, 2008. — 576 с.: ил. + CD-ROM

ISBN 978-5-9775-0157-6

Рассмотрены низкоуровневая (API-функции) и высокоуровневая (библиотека классов MFC) технологии прикладного программирования в среде в Microsoft Visual Studio C++ .NET для ОС Windows. Подробно описаны дочерние окна, редактор ресурсов, меню, панели инструментов, строка статуса, диалоговые окна и более 15 самых популярных управляющих элементов для них, динамические подключаемые библиотеки и мастера. Материал сопровождается демонстрационными примерами, вопросами и упражнениями для самопроверки с ответами, тестами и заданиями для курсового проектирования, которые также помещены на прилагаемом компакт-диске.

Для студентов, преподавателей технических вузов и программистов

УДК 681.3.068
ББК 32.973.26-018.1

Группа подготовки издания:

Главный редактор	*Екатерина Кондукова*
Зам. главного редактора	*Игорь Шишигин*
Зав. редакцией	*Григорий Добин*
Редактор	*Андрей Смышляев*
Компьютерная верстка	*Ольги Сергиенко*
Корректор	*Зинаида Дмитриева*
Дизайн серии	*Инны Тачиной*
Зав. производством	*Николай Тверских*

Лицензия ИД № 02429 от 24.07.00. Подписано в печать 25.12.07.
Формат 70×100 1/16. Печать офсетная. Усл. печ. л. 46,44.
Тираж 1500 экз. Заказ №
"БХВ-Петербург", 194354, Санкт-Петербург, ул. Есенина, 5Б.

Санитарно-эпидемиологическое заключение на продукцию
№ 77.99.02.953.Д.006421.11.04 от 11.11.2004 г. выдано Федеральной службой
по надзору в сфере защиты прав потребителей и благополучия человека.

Отпечатано с готовых диапозитивов
в ГУП "Типография "Наука"
199034, Санкт-Петербург, 9 линия, 12

ISBN 978-5-9775-0157-6

Оглавление

Предисловие

Эта книга, наряду с учебным пособием "Технологии программирования. C++" [1], изданным ранее в "БХВ-Петербург", обеспечивает курс "Технологии программирования" и соответствует разработанной, с участием автора, примерной программе этого курса, рекомендованной Министерством образования для подготовки бакалавров, магистров и инженеров по направлению 220200 "Автоматизация и управление" (региональная часть государственного образовательного стандарта). Поскольку курс "Технологии программирования" является продолжением курса "Программирование и основы алгоритмизации", то и книга, которую вы сейчас читаете, является продолжением учебного пособия по курсу "Программирование и основы алгоритмизации" [2], вышедшего в издательстве "Высшая школа". Пособие ориентировано на студентов, но может быть полезным и преподавателям высших учебных заведений, а также программистам, создающим программные продукты с использованием языка C++.

В книге рассмотрены вопросы, связанные с разработкой оконных приложений для операционных систем Windows с использованием низкоуровневых средств программирования: набор базовых интерфейсов программирования приложений для Windows — Application Programming Interface (API), и набор из средств разработки, утилит и документации, позволяющий создавать приложения для платформы Windows — Software Development Kit (SDK или Platform SDK). А также высокоуровневых средств разработки на основе библиотеки классов фирмы Microsoft — Microsoft Foundation Classes (MFC). Демонстрационные примеры в первых главах части книги выполнены в двух вариантах, с использованием Windows API и MFC. Это сделано для того, чтобы читатель мог сопоставить разные средства достижения одного и того же результата и детально разобраться в особенностях разработки оконных приложений. Все иллюстрирующие программы хорошо структурированы — в них последовательно выполнена файловая, функциональная и объектно-ориентированная декомпозиция.

В книге, применительно к IDE (Integrated Development Environment, интегрированная среда разработки) Microsoft Visual Studio 2005, последовательно рассмотрены следующие основные вопросы:

1. Базовые концепции программирования оконных приложений. Общая характеристика библиотеки классов MFC.
2. Создание простейших Windows-приложений. Контекст, вывод в главное окно текстовой и графической информации.
3. Создание Windows-приложений с дочерними окнами и кнопками. Обработка кнопок.
4. Редактор ресурсов. Создание Windows-приложений с поддержкой меню. Меню, команды, акселераторы, "горячие" клавиши. Таблица строк.
5. Редактор ресурсов. Создание приложений на базе MFC с панелями инструментов и строкой статуса. Всплывающие подсказки, сообщения для строки статуса, конфигурирование панелей инструментов и строки статуса.
6. Редактор ресурсов. Создание приложений на базе MFC с диалоговыми окнами. Стандартные и пользовательские диалоговые окна. Модальные и немодальные диалоговые окна. Использование управляющих элементов в диалоговых и обычных окнах (однострочный редактор, радиокнопки).
7. Редактор ресурсов. Создание приложений на базе MFC с диалоговыми окнами. Использование управляющих элементов в диалоговых окнах (группировка управляющих элементов, отмечаемые и растровые кнопки, полосы прокрутки и т. п.).
8. Создание и использование динамически подключаемых библиотек для приложений на базе MFC.
9. Создание каркаса приложения на базе MFC с помощью мастера MFC Application Wizard. Русификация полученного каркаса.
10. Модификация каркаса приложения на базе MFC, полученного с помощью мастера MFC Application Wizard. Добавление меню и панелей инструментов. Загрузка и выгрузка файлов в дочернее окно (или дочерние окна). Интеграция приложения в полученный каркас. Добавление новых тем в справочную систему и их русификация.

В приложениях к книге приведены следующие полезные сведения:

- ответы к вопросам для самопроверки;
- варианты тестов и заданий на курсовое проектирование;
- методика создания и отладки оконного приложения в IDE Microsoft Visual Studio 2005.

Для удобства читателей книга содержит более 120 вариантов тестовых контрольных заданий по основным разделам, 30 вариантов заданий на курсовое

проектирование и возможный пример тестовых экзаменационных вопросов. В книгу включено более 210 вопросов для самопроверки, снабженных ответами, и более 70 упражнений для самопроверки, что позволяет использовать ее и для *самостоятельного* изучения материала. Прилагаемый компакт-диск содержит файл с вариантами контрольных тестовых заданий по основным разделам и заданий на курсовое проектирование, а также полные исходные тексты демонстрационных программ, имеющихся в книге (36 демонстрационных примеров). Восемь демонстрационных примеров разработано совместно с проф. Лекаревым М. Ф. и часть их опубликована [7]. Из сказанного со всей очевидностью следует, что книга поддерживает не только лекционную часть учебного курса, но и полностью поддерживает практические занятия.

Используемые обозначения

Исходные коды программ и результаты их работы, приводимые в книге, для удобства читателей печатаются с использованием моноширинного шрифта `Courier New`. Названия окон, полей окон, меню, команд, ресурсов, клавиш и кнопок в тексте книги выделяются **полужирным шрифтом**.

Курсивом в тексте выделяются определяющие вхождения новых понятий, а также отдельные слова или выражения, на которые следует обратить внимание.

Имена файлов и их расширения пишутся без кавычек и без выделения.

Кроме шрифтовых выделений, используется три типа специальных абзацев: советы, замечания и примечания.

Совет

Всегда стремитесь использовать масштабируемые шрифты.

Замечание

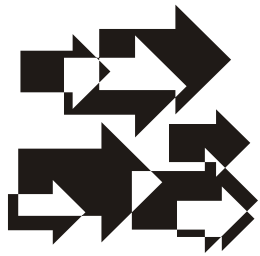
Чтобы вы могли эффективно общаться и легко понимать друг друга, внимательно изучите все описанные далее термины и понятия.

Примечание

Наряду с перечисленными компонентами окна имеются и другие компоненты, такие как: панели инструментов с кнопками, строка статуса, управляющие элементы (всевозможные кнопки, элементы редактирования, различные списки, индикаторы прогресса и т. п.), которые будут рассмотрены далее.

Ваши отзывы о книге, конструктивные замечания и критику направляйте по адресу: **davydov@aivt.ftk.spbstu.ru**.

ГЛАВА 1



Базовые концепции программирования [3]

Особенность создания Windows-приложений, т. е. приложений, предназначенных для работы под управлением операционной системы (ОС) Windows (далее Windows), заключается в том, что здесь применяются специальные методики программирования и используется своя терминология, которую можно разделить на две большие категории:

- терминология, связанная с пользовательским интерфейсом (меню, диалоговые окна, пиктограммы и т. д.);
- терминология, относящаяся непосредственно к программированию (сообщения, вызовы функций и т. д.).

Примечание

Чтобы вы могли эффективно общаться и легко понимать друг друга, внимательно изучите все описанные далее термины и понятия.

1.1. Что представляет собой окно?

Окно — это специальная прямоугольная область экрана. Все элементы окна, его размер и внешний вид контролируются открывающей его программой. Каждый щелчок мышью, выполняемый пользователем на каком-либо элементе окна, вызывает ответные действия приложения. Многозадачность в Windows заключается, в частности, в возможности одновременного открытия окон нескольких приложений или же нескольких окон одного приложения. Активизируя с помощью мыши или клавиатуры то или иное окно, пользователь дает системе понять, что последующие команды и данные следует направлять именно этому окну.

1.2. Компоненты окна

Стандартный внешний вид окон Windows-приложений и предсказуемость работы различных их *компонентов* позволяют пользователям чувствовать себя уверенно с новыми приложениями и легко разбираться в принципах их работы. Основные компоненты любого окна показаны на рис. 1.1.

Рис. 1.1. Основные компоненты окна

1.2.1. Рамка

Обычно каждое окно заключается в небольшую *рамку*. Новичку может показаться, что функции рамки сводятся только к отделению окна от остальных частей экрана. Но в действительности это не так. Рамка окна, как правило, является и средством *масштабирования*. Размер окна приложения при его соответствующем стиле можно изменить. Для этого достаточно поместить указатель мыши на рамку и перетащить его, удерживая нажатой левую кнопку мыши.

1.2.2. Строка заголовка

Имя приложения, которому принадлежит открытое окно, отображается в *строке заголовка*, в верхней части окна. Строка заголовка является обяза-

тельным элементом всех окон приложений и позволяет пользователю легко определить, какому приложению принадлежит конкретное окно, если в системе запущено одновременно несколько приложений. Строка заголовка активного окна выделяется альтернативным цветом, чтобы активное окно легко можно было отличить от неактивных окон.

1.2.3. Значок (пиктограмма) приложения

Другим обязательным элементом любого окна является расположенный в его левом верхнем углу *значок приложения*. Этот значок обычно представляет собой маленький логотип приложения. Щелчок на значке приводит к открытию системного меню.

1.2.4. Системное меню

В *системном меню* представлены стандартные команды управления окном. Например, для локализованной, русскоязычной ОС: **Восстановить**, **Переместить**, **Размер**, **Свернуть**, **Развернуть** и **Закрыть** (рис. 1.2).

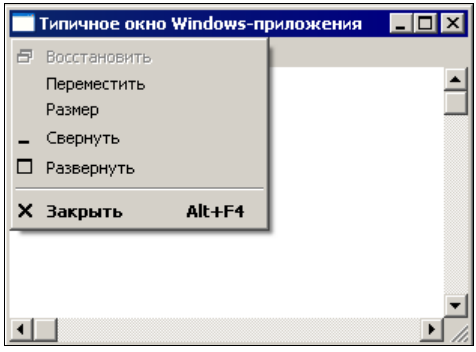


Рис. 1.2. Системное меню

1.2.5. Кнопка свертывания

В правом верхнем углу большинства окон приложений имеются три кнопки. Крайняя левая из них предназначена для *свертывания окна* в пиктограмму на панели задач. Такой же результат можно получить с помощью команды **Свернуть** системного меню.

1.2.6. Кнопка развертывания/восстановления

Средняя кнопка в правом верхнем углу либо *разворачивает окно* на весь экран, либо *восстанавливает* его прежние размеры, если окно уже развернуто.

Такой же результат можно получить с помощью команд **Развернуть** и **Восстановить** системного меню.

1.2.7. Кнопка закрытия

Крайняя правая кнопка в правом верхнем углу предназначена для *закрытия приложения*. После закрытия окна активным автоматически становится окно следующего приложения. Закрыть окно можно также с помощью команды **Закрыть** системного меню.

1.2.8. Вертикальная полоса прокрутки

В некоторых случаях окно приложения может содержать *вертикальную полосу прокрутки*, которая располагается по правому краю окна. В верхней и нижней частях полосы находятся кнопки со стрелками, направленными вверх и вниз соответственно. Вдоль самой полосы располагается бегунок. Положение бегунка показывает, какая часть окна или документа отображается в данный момент на экране. Перетаскивая бегунок с помощью мыши или клавиатуры, можно выбрать нужную часть многостраничного документа. Щелчок мышью на кнопке со стрелкой приведет к смещению содержимого окна на одну строку вверх или вниз, а щелчок на свободном пространстве выше или ниже бегунка — на одну экранную страницу вверх или вниз.

1.2.9. Горизонтальная полоса прокрутки

Окно может быть также оснащено *горизонтальной полосой прокрутки*, которая располагается по нижнему краю окна и работает аналогично вертикальной полосе прокрутки. Горизонтальная полоса прокрутки предназначена для выведения на экран частей документов, состоящих из большого числа столбцов. Щелчок мышью на кнопках со стрелками приведет к смещению содержимого окна на один столбец влево или вправо. Щелчок на областях между кнопками со стрелками и бегунком смещает изображение на одну экранную страницу влево или вправо.

1.2.10. Строка меню

В большинстве приложений, под строкой заголовка, находится *строка меню*, содержащая наборы команд и опций программы. Обычно для выбора команд меню используется мышь, но эти действия можно выполнить и с помощью клавиатуры. Каждому элементу меню, как правило, соответствует *клавиша быстрого вызова* ("горячая" клавиша), выделенная подчеркиванием в названии элемента. Чтобы вызвать данный элемент, нужно последовательно нажать клавишу <Alt> и соответствующую клавишу быстрого вызова. Так, по-

следовательное нажатие клавиш <Alt> и <F> для окна, представленного на рис. 1.1, открывает меню **File** (Файл). С помощью встроенного в IDE редактора ресурсов вы можете создавать собственные меню.

1.2.11. Рабочая область

Рабочая область обычно занимает большую часть окна. Именно в эту область программа выводит результаты своей работы.

Примечание

Наряду с перечисленными компонентами окна имеются и другие компоненты, такие как панели инструментов с кнопками, строка статуса, управляющие элементы (всевозможные кнопки, элементы редактирования, различные списки, индикаторы прогресса и т. п.), которые будут рассмотрены далее.

1.3. Классы окон

Чтобы два окна выглядели и работали совершенно одинаково, они оба должны базироваться на общем *классе окна*. В приложениях класс окна регистрируется программой в процессе инициализации. Зарегистрированный класс становится доступным для всех программ, запущенных в данный момент в системе. Как мы увидим далее, в случае использования библиотеки классов MFC, вся работа по регистрации классов окон выполняется автоматически, что существенно облегчает работу программиста.

Благодаря тому, что окна приложения создаются на основе общего базового класса, значительно сокращается объем информации, которую при этом следует указывать. Поскольку класс окна содержит в себе описания элементов, общих для всех окон данного класса, нет необходимости повторять эти описания при создании каждого нового окна. К тому же в приложениях, использующих функции API, все окна одного класса используют одну общую оконную процедуру (функцию), обеспечивающую работу с различными однотипными окнами. Это позволяет избежать дублирования кода.

1.4. Графические объекты, используемые в окнах

Примерами *графических объектов*, с которыми можно обращаться как с единым целым и которые выступают элементами пользовательского интерфейса, являются: строка меню, кнопки, полосы прокрутки и т. д. Другие, широко используемые графические объекты оконных приложений Windows кратко будут описаны далее.

1.4.1. Значки

Значками называются маленькие графические изображения, выполняющие опознавательную функцию. Так, значки приложений на панели задач позволяют легко определить, какие программы в настоящий момент запущены, даже если названия программ не отображаются целиком. Значки могут быть полезны и в самих приложениях, поскольку с их помощью можно привлекать внимание пользователей к сообщениям об ошибках и различным предупреждениям. В состав операционной системы Windows входит набор стандартных значков, в частности, стилизованные знак вопроса, восклицательный знак и ряд других значков. С помощью встроенного в IDE редактора ресурсов вы можете создавать собственные значки.

1.4.2. Указатели мыши

Указатели мыши также являются графическими объектами, используемыми для отслеживания перемещения мыши. Вид указателя может меняться в зависимости от выполняемого задания и состояния системы. Например, стандартный указатель в виде стрелки изменяет свой вид на изображение песочных часов в том случае, если система занята. С помощью встроенного в IDE редактора ресурсов вы можете создавать собственные указатели мыши.

1.4.3. Текстовые курсоры

Курсоры предназначены для указания места, куда следует осуществлять ввод текстовых данных. Отличительной особенностью курсоров является их мерцание. В большинстве текстовых редакторов и полях диалоговых окон в качестве курсора применяется курсор в виде символа "I". Обратите внимание, что в Windows нет (в отличие от значков и указателей мыши) коллекции готовых курсоров.

1.4.4. Окна сообщений

Окна сообщений представляют собой разновидность диалоговых окон, содержащих строку заголовка, значок (значки), текст сообщения и кнопку (кнопки). На рис. 1.3 показано стандартное окно сообщения, которое появляется при закрытии окна программы Notepad (Блокнот) в том случае, если в нем содержатся несохраненные данные.

Окно сообщений создается путем вызова функции `MessageBox`, аргументы которой задают текст заголовка окна, текст сообщения, какой из стандартных значков Windows использовать (если это необходимо) и какой набор кнопок

выводить. В частности, можно вызывать окна со следующими комбинациями кнопок: **Abort**/**Retry**/**Ignore** (Прервать/Повторить/Игнорировать), **OK**, **Yes**/**No** (Да/Нет), **Yes**/**No**/**Cancel** (Да/Нет/Отмена), **OK**/**Cancel** (ОК/Отмена) и **Retry**/**Cancel** (Повторить/Отмена).

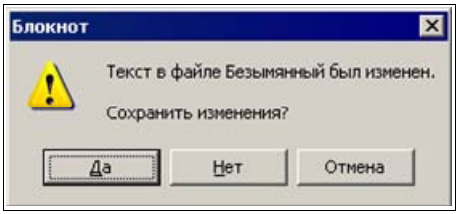


Рис. 1.3. Окно сообщения текстового редактора Notepad

1.4.5. Диалоговые окна

Диалоговые окна содержат наборы различных элементов управления. Они позволяют пользователю задавать опции и параметры программы, которой принадлежит диалоговое окно. Пример диалогового окна для настройки параметров печати показан на рис. 1.4. Windows автоматически отображает элементы управления, содержащиеся в окне. Внешний вид диалогового окна разрабатывается с помощью встроенного в IDE редактора ресурсов.

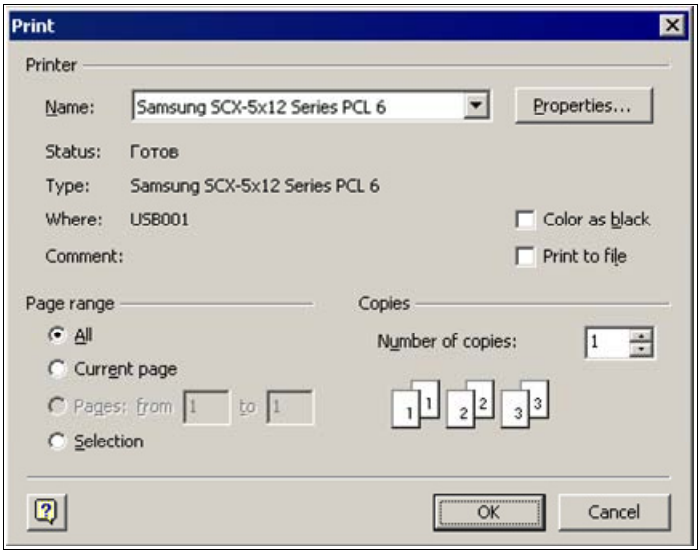


Рис. 1.4. Диалоговое окно настройки параметров печати

1.4.6. Шрифты

Шрифт в Windows — это графический ресурс, содержащий набор символов определенного типа. Шрифты бывают растровые, контурные и масштабируемые.

Растровый шрифт (raster font) представляет символы с помощью растровых изображений. Такие изображения плохо поддаются масштабированию, из-за чего страдает качество образов.

Контурный шрифт (stroke font) представляет символы с помощью линий. Такие шрифты предназначены, прежде всего, для графопостроителей (плоттеров).

Масштабируемый шрифт TrueType, (true type font) представляет символы с помощью линий и сплайновых кривых. Такие шрифты масштабируются практически до любого размера без ухудшения качества образов.

Примечание

Всегда стремитесь использовать масштабируемые шрифты.

Существует набор функций, с помощью которых можно манипулировать начертанием символов для получения форматированного текста. В приложениях можно использовать как стандартные шрифты, так и пользовательские шрифты. Встроенные функции позволяют получать, на базе основного шрифта, полужирное начертание, курсив, подчеркнутый текст и изменять размер шрифта. Внешний вид шрифта можно сделать независимым от типа устройства, на которое выводится текст. Так, при использовании технологии TrueType обеспечивается соответствие между внешним видом шрифта на экране и шрифта, выводимого при печати.

1.4.7. Точечные рисунки

Точечные рисунки представляют собой точную копию части экрана, снятую попиксельно. Тот факт, что изображение является точным образом экрана, устраняет необходимость в каких-либо дополнительных преобразованиях, что существенно сокращает время вывода изображения на экран. В Windows точечные рисунки наиболее широко применяются для двух целей. Во-первых, они служат изображениями всевозможных кнопок и значков, например, стрелок полос прокрутки и кнопок панелей инструментов. Другой областью применения точечных рисунков являются кисти, с помощью которых рисуются и заполняются цветом различные геометрические фигуры на экране.

Точечные рисунки можно создавать и модифицировать с помощью встроенного редактора ресурсов.

1.4.8. Перья

Перья предназначены для рисования геометрических фигур и различных контуров. Перья характеризуются тремя основными параметрами:

- ❒ шириной линии;
- ❒ стилем (точечный, штрихпунктирный, непрерывный);
- ❒ цветом.

Существует два готовых пера: одно для рисования черных линий, другое — для рисования белых. С помощью специальных функций вы можете создавать собственные перья.

1.4.9. Кисти

Кисти предназначены для заливки объектов цветом, выбранным из заданной палитры. Минимальный размер кисти — 8×8 пикселов. Кисть также характеризуется тремя параметрами:

- ❒ размером;
- ❒ шаблоном заливки;
- ❒ цветом.

Заливка может быть сплошной, штриховой, диагональной или представлять собой узор, заданный пользователем.

1.5. Принципы обработки сообщений

Ни одно приложение в Windows не отображает свои данные непосредственно на экране, не обрабатывает напрямую прерывания устройств и не выводит данные непосредственно на печать. Вместо всего этого приложение вызывает встроенные функции Windows и ожидает от системы соответствующих сообщений.

Операционная система Windows является *событийно-управляемой* ОС. Этот термин означает, что отдельные части ОС взаимодействуют между собой, а также с прикладными программами посредством сообщений.

Сообщение — это форма регистрации событий в операционной системе (или просто в системе). Система использует сообщения для того, чтобы сигнализировать о совершении событий.

Подсистема сообщений в Windows, используемая в операционной системе и пользовательских приложениях, — это средство распределения информации в многозадачной среде. С точки зрения приложения, сообщение — это уве-

домление о некотором произошедшем событии, на которое приложение должно отреагировать определенным образом. Такое событие может быть инициировано пользователем, например, нажатием клавиши или перемещением мыши, изменением размера окна или выбором команды из меню. Но события могут порождаться и самим приложением.

Особенность этого процесса состоит в том, что приложение должно быть полностью ориентировано на прием и обработку сообщений. Программа должна быть готова в любой момент принять сообщение, определить его тип, выполнить соответствующую обработку и вновь перейти в режим ожидания до поступления следующего сообщения.

Приложения ОС Windows существенно отличаются от приложений, написанных для MS DOS. ОС Windows открывает приложениям доступ к сотням встроенных функций, которые можно вызывать напрямую (низкий уровень) или косвенно (высокий уровень), посредством библиотек типа MFC. Эти функции содержатся в ряде модулей, таких как KERNEL, GDI и USER. Функции модуля KERNEL отвечают за управление памятью, загрузку приложений, выполнение приложений и распределение системных ресурсов. Модуль GDI содержит функции создания и отображения графических объектов, а модуль USER отвечает за выполнение всех других функций, обеспечивающих взаимодействие приложений с пользователями и средой ОС Windows.

1.5.1. Формат сообщений

Сообщения используются для информирования приложения о том, что в системе произошло то или иное событие. На практике, сообщения направляются не столько самому приложению, сколько определенному окну, открытому этим приложением.

Реально в Windows существует только один механизм обработки сообщений — системная очередь сообщений. Но каждое выполняющееся приложение организовывает и свою очередь. Функции модуля USER, в частности, ответственны за передачу сообщений из системной очереди в очередь конкретного приложения. Таким образом, в очереди конкретного приложения накапливаются сообщения, адресованные любому окну, открытому данным приложением.

Независимо от типа, все сообщения для 32-разрядных версий Windows характеризуются четырьмя параметрами (рис. 1.5):

- дескриптором окна, которому адресуется данное сообщение;
- типом сообщения;
- еще двумя 32-разрядными параметрами.

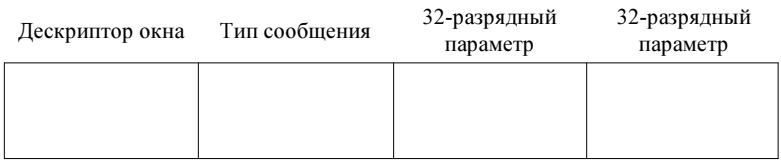


Рис. 1.5. Структура сообщения

Дескрипторы (handle, описатель) широко используются в приложениях ОС Windows. Дескриптором называется уникальный номер (беззнаковое 32-разрядное целое значение), который присваивается всем системным объектам, таким как: окна, элементы управления, меню, значки, перья и кисти, а также областям памяти, устройствам вывода и т. д.

Поскольку Windows позволяет одновременно открывать несколько копий одного приложения, операционная система должна иметь возможность отслеживать каждую копию в отдельности. Это достигается путем присвоения каждому экземпляру программы своего дескриптора.

Дескрипторы обычно служат в качестве индексов системной таблицы объектов. Благодаря тому, что доступ к объектам осуществляется по индексам таблицы, а не по их непосредственным адресам в памяти, Windows может динамически перераспределять ресурсы за счет обновления адресов в таблице. Например, если Windows связала некоторый ресурс приложения с 16-й строкой таблицы, то независимо от того, куда впоследствии Windows переместит этот ресурс, его текущий адрес всегда будет представлен в 16-й строке.

Тип сообщения (второй параметр) задается идентификатором, который определен в одном из файлов заголовков Windows. Для работы с идентификаторами сообщений в программу включается файл windows.h. Как правило, идентификаторы начинаются с двухсимвольного префикса, за которым следует символ подчеркивания. Так, оконные сообщения начинаются с префикса `WM_`: `WM_CREATE`, `WM_DESTROY`, `WM_SIZE`, `WM_PAINT`, `WM_QUIT`, `WM_COMMAND` и др. Сообщения кнопок имеют префикс `BM_`, полей — `EM_` и т. д. В приложении можно также создать и зарегистрировать собственный тип сообщения, предназначенного для частных целей.

Последние два параметра сообщения несут дополнительную информацию. Их содержание может изменяться в зависимости от типа сообщения. Например, посредством этих параметров может передаваться информация о том, какая клавиша была нажата, какая команда меню активизирована и т. д.

Форматы некоторых оконных сообщений приведены на рис. 1.6—1.10. Разберем их.

Система посылает окну Windows сообщение `WM_CREATE` *после* его создания, но *до* появления образа окна на экране. Свой первый параметр сообщение

WM_CREATE не использует. Второй параметр содержит указатель на структуру типа CREATESTRUCT, которая содержит информацию о создаваемом окне. Если приложение обрабатывает WM_CREATE, то при успешной работе оно должно возвращать 0. При этом создание окна продолжается. Если же приложение возвращает –1, то функция создания окна (CreateWindow или CreateWindowEx) возвращает пустой манипулятор (со значением 0).

Рис. 1.6. Структура сообщения WM_CREATE

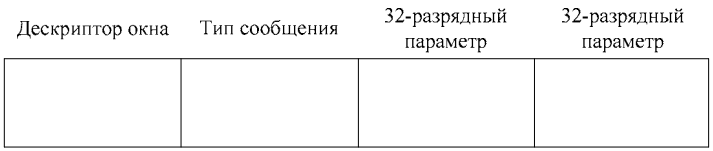


Рис. 1.7. Структура сообщения WM_DESTROY

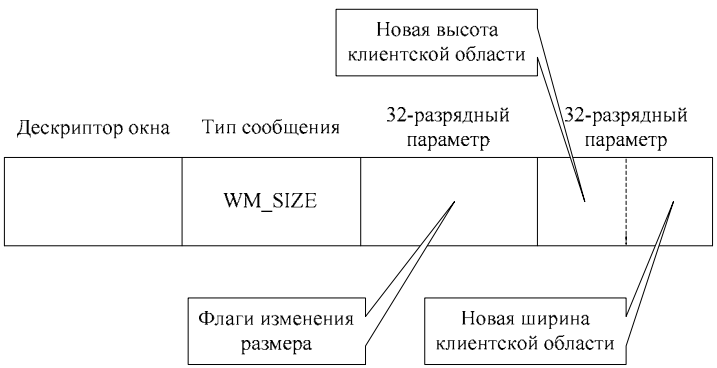


Рис. 1.8. Структура сообщения WM_SIZE

Система посылает сообщение WM_DESTROY окну перед его уничтожением *после* того, как образ окна удален с экрана. Сообщение WM_DESTROY не использует

оба параметра. Если приложение обрабатывает `WM_DESTROY`, то оно должно возвращать значение 0.

Система посылает сообщение `WM_SIZE` окну Windows *после* того, как размер окна изменен. Значения новых размеров клиентской области окна можно получить с помощью макросов: `LOWORD(lParam)` дает ширину, а `HIWORD(lParam)` — высоту. Если приложение обрабатывает `WM_SIZE`, то оно должно возвращать значение 0.

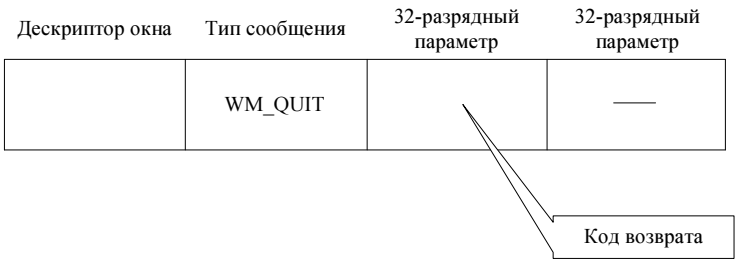


Рис. 1.9. Структура сообщения `WM_QUIT`

Сообщение `WM_QUIT` означает запрос на завершение работы приложения. Оно приводит к тому, что специальная функция `GetMessage` возвращает значение 0. Первый параметр задает значение кода возврата, который приложение передает ОС при своем завершении.

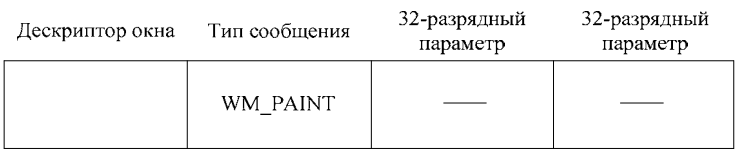


Рис. 1.10. Структура сообщения `WM_PAINT`

Сообщение `WM_PAINT` посылается окну, если система, или другое приложение, выдает запрос на перерисовку окна (или его части). В частности, `WM_PAINT` посылается в результате вызова функций `UpdateWindow` и `RedrawWindow`. Сообщение `WM_PAINT` не использует оба параметра. Если приложение обрабатывает `WM_PAINT`, то оно должно возвращать значение 0.

1.5.2. Генерирование сообщений

Именно реализация концепции обмена сообщениями позволяет Windows быть многозадачной средой. Работа Windows основана на передаче, приеме и обработке сообщений.

Существует четыре основных *источника*, от которых приложение может получить сообщение:

- пользователь;
- ОС Windows;
- само приложение;
- другие приложения.

Сообщения пользователей включают информацию о вводе текста, перемещении мыши, нажатии кнопок мыши, выборе команд меню, перемещении бегунка полос прокрутки и т. д. Большую часть времени приложение занято обработкой именно таких сообщений. Получение сообщения от пользователя означает, что человек, запустивший программу, хочет изменить ход ее выполнения.

Системные сообщения посылаются программе при изменении ее состояния. Например, щелчок на значке приложения означает, что пользователь хочет сделать данное приложение активным. В этом случае Windows сообщает приложению, что на экране открывается его окно, размер и положение окна изменяются и т. д. В зависимости от текущего состояния приложения сообщение от Windows может быть принято и обработано либо проигнорировано.

В следующем разделе мы рассмотрим, как создаются простейшие приложения Windows. Вы узнаете, что, по сути, в приложении используется ряд процедур, каждая из которых отвечает за обработку определенного типа сообщения для определенного окна. Например, одна из таких процедур может отвечать за изменение размеров окна. При этом совсем не обязательно, чтобы это происходило только по команде пользователя, — решение об изменении может принимать и сама программа.

Сообщения от других приложений в настоящее время применяются достаточно редко. Примером межзадачного обмена сообщениями может служить протокол DDE (Dynamic Data Exchange, динамический обмен данными).

1.5.3. Обработка сообщений

Оконные приложения Windows, написанные на языке C++, содержат специальные процедуры для обработки всех типов сообщений, которые могут быть посланы программе. Разные окна программы могут по-разному реагировать на одни и те же сообщения. Это достигается за счет того, что *обработчики сообщений* пишутся отдельно для каждого окна, а Windows знает, какому именно окну адресовано сообщение. Таким образом, приложения содержат процедуры обработки не только сообщений разных типов, но и сообщений для разных окон.

1.5.4. Цикл обработки сообщений

Основным компонентом любого приложения ОС Windows является *цикл обработки сообщений*. Если говорить в целом, то работа приложения организуется следующим образом. Сначала приложение создает и открывает свои окна (в общем случае их может быть несколько), затем запускается цикл обработки сообщений и, в конце концов, при получении сообщения `WM_CLOSE`, работа приложения завершается. Цикл обработки сообщений ответственен за передачу поступающих от Windows сообщений соответствующим оконным процедурам.

На последовательность обработки сообщений влияют два фактора:

- организация очереди, в которую помещается сообщение;
- приоритет приложения.

Сообщения могут поступать из двух очередей — *системной* и очереди *приложения*. Важным является то, что, даже если сообщение поступило от самого приложения, сначала оно будет помещено в системную очередь. Когда в системной очереди подойдет черед сообщения, оно будет направлено в очередь соответствующего приложения. Такая организация работы системной очереди позволяет Windows контролировать прохождение всех сообщений и ограничивает ответственность приложений обработкой только тех сообщений, которые относятся непосредственно к приложениям.

Сообщения обычно помещаются в очередь по принципу FIFO (First In, First Out, первым поступило, первым обслужено). Такие сообщения называются *синхронными*. Но иногда Windows вставляет сообщение сразу в голову очереди. Сообщения такого типа, изменяющие нормальный ход выполнения программы, называются *асинхронными*.

Существует несколько видов асинхронных сообщений, среди них — сообщение о перерисовке, сообщение таймера и сообщение о завершении программы. Например, сообщение таймера может запускать некоторые действия в определенный момент времени, независимо от того, какие сообщения сейчас находятся в очереди; оно имеет наивысший приоритет и передается раньше всех других сообщений.

Возникает вопрос — как Windows выходит из положения, если сообщение предназначено одновременно нескольким приложениям? Эта коллизия разрешается одним из двух способов. Первый состоит в задании *приоритетов*. Когда загружается некоторое приложение, для него автоматически устанавливается нулевой приоритет. В процессе работы приложения его приоритет может быть изменен. Любые конфликты разрешаются в пользу того приложения, чей приоритет выше.

Однако изменять приоритет приложения, заданный по умолчанию, обычно не рекомендуется. Тем более что в распоряжении Windows есть еще один метод,