[3. Начало работы с интегрированной средой разработки.](#_Toc479802287)

[3.1. Запуск 32-разрядной среды Visual C++.](#_Toc479802288)

[3.2. Меню File (файл).](#_Toc479802289)

[3.2.1. New...](#_Toc479802290)

[3.2.2. Open...](#_Toc479802291)

[3.2.3. Close.](#_Toc479802292)

[3.2.4. Open Workspace.](#_Toc479802293)

[3.2.5. Close Workspace.](#_Toc479802294)

[3.2.6. Save.](#_Toc479802295)

[3.2.7. Save As...](#_Toc479802296)

[3.2.8. Save All.](#_Toc479802297)

[3.2.9. Find in Files...](#_Toc479802298)

[3.2.10. Page Setup...](#_Toc479802299)

[3.2.11. Print...](#_Toc479802300)

[3.2.12. Список последних проектов.](#_Toc479802301)

[3.2.13. Exit.](#_Toc479802302)

[3.3. Меню Edit.](#_Toc479802303)

[3.3.1. Undo.](#_Toc479802304)

[3.3.2. Redo.](#_Toc479802305)

[3.3.3. Cut.](#_Toc479802306)

[3.3.4. Сору.](#_Toc479802307)

[3.3.5. Paste.](#_Toc479802308)

[3.3.6. Delete.](#_Toc479802309)

[3.3.7. Select All.](#_Toc479802310)

[3.3.8. Find...](#_Toc479802311)

[3.3.9. Replace...](#_Toc479802312)

[3.3.10. Go To...](#_Toc479802313)

[3.3.11. InfoViewer Bookmarks...](#_Toc479802314)

[3.3.12. Bookmark.](#_Toc479802315)

[3.3.13. Breakpoints...](#_Toc479802316)

[3.3.14. Properties...](#_Toc479802317)

[3.4. Меню View.](#_Toc479802318)

[3.4.1. ClassWizard...](#_Toc479802319)

[3.4.2. Resource Symbols...](#_Toc479802320)

[3.4.3. Resource Includes...](#_Toc479802321)

[3.4.4. Full Screen.](#_Toc479802322)

[3.4.5. Toolbars...](#_Toc479802323)

[3.4.6. InfoViewer Query Results.](#_Toc479802324)

[3.4.7. InfoViewer History List.](#_Toc479802325)

[3.4.8. Project Workspace.](#_Toc479802326)

[3.4.9. Info Viewer Topic.](#_Toc479802327)

[3.4.10. Output.](#_Toc479802328)

[3.4.11. Watch.](#_Toc479802329)

[3.4.12. Variables.](#_Toc479802330)

[3.4.13. Registers.](#_Toc479802331)

[3.4.14. Memory.](#_Toc479802332)

[3.4.15. Call Stack.](#_Toc479802333)

[3.4.16. Disassembly.](#_Toc479802334)

[3.5. Меню Insert.](#_Toc479802335)

[3.6. Меню Build.](#_Toc479802336)

[3.6.1. Compile.](#_Toc479802337)

[3.6.2. Build.](#_Toc479802338)

[3.6.3. Rebuild All.](#_Toc479802339)

[3.6.4. Batch Build...](#_Toc479802340)

[3.6.5. Stop Build.](#_Toc479802341)

[3.6.6. Update All Dependencies.](#_Toc479802342)

[3.6.7. Debug.](#_Toc479802343)

[3.6.8. Execute.](#_Toc479802344)

[3.6.9. Settings...](#_Toc479802345)

[3.6.10. Configurations...](#_Toc479802346)

[3.6.11. Subprojects...](#_Toc479802347)

[3.6.12. Set Default Configuration...](#_Toc479802348)

[3.7. Меню Tools.](#_Toc479802349)

[3.7.1. Browse...](#_Toc479802350)

[3.7.2. Close Browse Info File.](#_Toc479802351)

[3.7.3. OLE Control Test Container.](#_Toc479802352)

[3.7.4. OLE Object View.](#_Toc479802353)

[3.8. Меню Window.](#_Toc479802354)

[3.8.1. New Window.](#_Toc479802355)

[3.8.2. Split.](#_Toc479802356)

[3.8.3. Hide.](#_Toc479802357)

[3.8.4. Cascade.](#_Toc479802358)

[3.8.5. Tile Horizontally, Tile Vertically.](#_Toc479802359)

[3.8.6. Close All...](#_Toc479802360)

[3.8.7. Windows.](#_Toc479802361)

[3.9. Меню Help.](#_Toc479802362)

1. Начало работы с интегрированной средой разработки.

Среда разработки, как 32-разрядная (версия 4.0), так и 16-разрядная (версия 2.0), представляет собой интегрированные в единую оболочку средства, позволяющие легко создавать, открывать, просматривать, редактировать, сохранять, транслировать и отлаживать все ваши программы на С и C++. Среда Visual C++ имеет также возможности настраивать рабочее место в соответствии с вашими предпочтениями и требованиями, накладываемыми Аппаратурой.

## Запуск 32-разрядной среды Visual C++.

Перед обсуждением конкретных возможностей Visual C++, рассмотрим некоторые общие для всех меню свойства. Например, до каждого пункта меню можно добраться двумя путями. Самый распространенный — поместить курсор мыши на нужный пункт и нажать левую кнопку. Второй путь — использовать подчеркнутый символ в названии пункта. Вы можете, к примеру, обратиться к меню File (файл), нажав клавишу <ALT> одновременно с символом <F>.

Команды меню можно выполнять при помощи описанных выше процедур, но часто есть еще один способ это сделать. Вы можете выполнять команды меню из любого места среды, нажав специально назначенную комбинацию клавиш (такую комбинацию в дальнейшем будем называть горячей клавишей). Если для данного пункта меню такая возможность есть, обозначение горячей клавиши будет приведено справа от названия пункта меню. Например, первый пункт меню File (файл) называется New... (создать).

Эту команду можно выполнить непосредственно, без обращения к меню, нажав комбинацию <CTRL>+<N>.

Дополнительные замечания по поводу меню: во-первых, если название пункта изображено серым цветом, это значит, что соответствующая команда в данный момент невыполнима. Как правило, среда таким образом предупреждает вас, что не выполнено некоторое обязательное условие. Например, команда Save (сохранить) из меню File (файл) недоступна, если окно редактора пусто. Среда знает, что нельзя сохранить то, чего не существует, и уведомляет вас об этом, изображая команду серым цветом.

Во-вторых, название пункта, заканчивающееся многоточием (...), указывает на то, что при выборе команды появляется окно диалога. Например, если вы выберете команду Open... (открыть) из меню File (файл), появится окно диалога Open (открыть файл). Наконец, вы можете выполнить некоторые команды меню, нажав соответствующую кнопку панели инструментов, которая располагается ниже строки меню.

Рассмотрим теперь некоторые полезные возможности среды, доступные посредством меню.

## Меню File (файл).

В меню File среды Visual C++ собраны стандартные средства работы с файлами, встречающиеся во многих приложениях Windows. На рис. 2.2 показаны команды, доступные через меню File.

### New...

Пункт New... (создать) открывает новое окно редактирования файла. Обычно с этого начинается создание программы. Среда автоматически называет и нумерует открытые вами окна. Нумерацияначинается с 1, поэтому заголовок первого окна всегда будет хxx/, второго окна — ххх2, и так Далее, ххх соответствует типу файла, с которым вы работаете (текст, проект, ресурс, рисунок, код, значок или курсор).

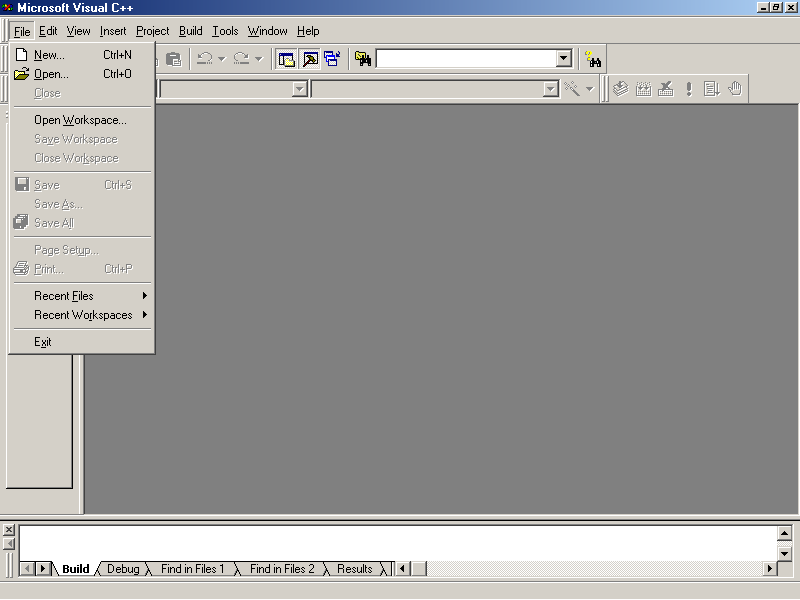


Рис. 2.2. Меню File среды Visual C++

Если вы открыли окна с xxx1 по ххх6, а затем решили закрыть окно ххх2, то в следующий раз, когда вы выполните команду New..., это название (в нашем случае ххх2) повторно использовано не будет. Windows автоматически выберет следующий по порядку номер (в нашем примере это будет ххх 7).

Самый быстрый способ вызвать окно диалога редактирования — щелкнуть на кнопке слева на панели инструментов. На этой кнопке изображен листочек бумаги. Нажатие этой кнопки эквивалентно выбору пункта New... в меню.

### Open...

В отличие от New..., который открывает окно редактирования не существовавшего ранее файла, пункт Open... (открыть)вызывает появление окна диалога с запросом информации о файле, который уже сохранен на диске. Это стандартное окно диалога Open File (открыть файл), в котором показаны текущее устройство, каталог и шаблон поиска файлов, и предлагается ввести нужные параметры.

В этом окне имеется полезное средство — автоматическое запоминание ваших настроек для использования в следующий раз, когда вы вызовете Open.Попытка открыть уже открытый файл автоматически вызывает звуковой сигнал и предупреждающее сообщение. Это полезное напоминание застрахует вас от возможности случайно открыть несколько копий одного и того же файла, отредактировать одну из них, а затем сохранить другую (не изменившуюся) копию. На панели инструментов вторая слева кнопка, где изображена открытая папка со стрелочкой, позволяет непосредственно выполнить команду Open.

### Close.

Команда Close (закрыть) используется, чтобы закрыть открытый файл. Если у вас открыто несколько файлов, будет закрыто активное (текущее) окно. Вы можете отличить активное окно от неактивного, посмотрев на рамку окна. Активное (текущее) окно имеет фокус ввода и изображается цветом, установленным в системе для отображения активных окон. Обычно для активных окон установлены выделенный цветом заголовок и темная рамка. Неактивные окна имеют обычно серые заголовок и рамку.

Если вы случайно попытаетесь закрыть не сохраненный файл, ничего страшного не случится. Интегрированная среда автоматически защитит вас от нежелательного развития событий, предупредив, что файл еще не сохранен, и спросит, не хотите ли вы его сохранить.

### Open Workspace.

Команда Open Workspace (открыть рабочее пространство)используется для активизации ранее сохраненного рабочего пространства. Рабочие пространства соответствуют приложениям, которые вы можете создавать. Проект состоит из одного набора исходных файлов и набора из одной или более конфигураций. Каждая конфигурация проекта вместе с набором файлов однозначно определяют двоичный файл, который в результате будет генерироваться.

### Close Workspace.

Команда Close Workspace (закрыть рабочее пространство)закрывает активное рабочее пространство. Это позволяет открыть другое рабочее пространство и перейти к работе над другим приложением.

### Save.

Команда Save (сохранить)записывает содержимое активного окна в соответствующий файл. Вы можете определить, существует ли такой файл, просто взглянув на заголовок окна. Если вы увидите автоматически созданный заголовок (вроде ххх1), значит, содержимое окна еще ни разу не было сохранено, и ему не назначено имени файла. При сохранении такого файла будет вызвано окно диалога Save As (сохранить как...). Вы также можете сохранить файлы, нажав соответствующую кнопку панели инструментов. Она расположена третьей слева, и на ней изображена Дискета. Если же файл был открыт в режиме "только для чтения" (см. описание команды EditProperties), рисунок будет серым, указывая на то, что возможность сохранить файл в данный момент недоступна.

### Save As...

Команда Save As... (сохранить как)позволяет создать копию содержимого активного окна под другим именем. Если вам не очевидно, зачем это может понадобиться, представьте себе такую ситуацию. Вы закончили работу над проектом. У вас есть работающая программа, но вы все же хотите попробовать внести пару изменений. Ради собственного спокойствия вы не хотите трогать работающую версию. Воспользовавшись командой Save As, вы можете сохранить файл под другим именем, и затем работать с его копией. Если новая версия работать не будет, можно вернуться к первоначальной версии файла.

### Save All.

Команда Save All (сохранить все)записывает в соответствующие файл, содержимое всех открытых окон. Если вы никогда не писали приложении Windows на С или C++, вас, возможно, ошеломит общее количество файлов, требуемых для создания работающей программы. Недостаток команды Save заключается в том, что она сохранит лишь содержимое активного окна.

Если какие-либо окна содержат не сохранявшийся ни разу текст, Save All автоматически вызовет для каждого из них окно диалога Save As, запрашивая ввод имени файла.

### Find in Files...

Команда Find in Files (поиск в файлах)позволяет найти последовательность символов в одном или нескольких файлах. Файлы, в которых следует искать, задаются указанием типа файла и каталога, где они находятся. Для задания искомой последовательности вы можете пользоваться регулярными выражениями. Результаты поиска отображаются в окне Output (вывод). Когда поиск закончен, можно открыть нужный файл, дважды щелкнув на его имени в окне Output.

### Page Setup...

Наиболее часто команда Page Setup (настройка параметров страницы)используется при оформлении распечаток текстов программ. Эта команда позволяет выбрать для каждой страницы верхний и нижний колонтитул, а также установить размеры полей. В табл. 2.1 приведены кодовые последовательности, используемые при оформлении колонтитулов.

Таблица 2.1.Кодовые последовательности

|  |  |
| --- | --- |
| **Код** | **Значение** |
| &c | Выровнять по центру |
| &d | Вставить текущую системную дату |
| &f | Вставить имя файла |
| &l | Выровнять влево |
| &p | Вставить номер страницы |
| &r | Выровнять вправо |
| &t | Вставить текущее системное время |

### Print...

Чтобы получить на бумаге копию содержимого активного окна, достаточно выполнить команду Print (печать). Окно диалога Print предоставляет вам несколько возможностей. Во-первых, вы можете решить, печатать ли все содержимое окна, или только выделенный текст. Для того чтобы выделить текст, поместите указатель мыши на первый символ выделяемого текста и, удерживая левую кнопку нажатой, переместите мышь вправо/вниз до конца выделяемого участка. После этого текст, который вы выделили, будет отображаться инверсным цветом (негативно). Если в окне есть выделенный текст, кнопка опции Print Range Selection (печатать только выделенный) будет изображена обычным цветом (не серым), указывая на то, что данная возможность доступна.

списками) состоит в том, что они зависят от контекста. Хронологические списки экономят ваше время, запоминая ранее задававшиеся значения какого-либо параметра. В этом меню список содержит имена ранее открывавшихся файлов. При первом запуске Visual C++ этот список пуст, поскольку файлы еще не открывались. Меню на рис. 2.2, показанном выше, содержит в хронологическом списке четыре файла.

### Список последних проектов.

Список последних использовавшихся проектов расположен в меню File непосредственно под хронологическим списком файлов. Список проектов схож со списком файлов, но содержит только имена проектов. Чтобы открыть любой файл из этих списков, щелкните левой кнопкой мыши на нужном имени. Меню на рис. 2.2, показанном выше, содержит в списке четыре имени проекта.

### Exit.

Команда Exit (выход)позволяет завершить работу среды Visual C++. Не волнуйтесь, если вы забыли сохранить какие-либо файлы; среда автоматически выдаст предупреждение и даст возможность сохранить все измененные файлы.

## Меню Edit.

Команды меню Edit (правка)позволяют быстро находить и исправлять текст в активном окне примерно так же, как это делается любым из распространенных текстовых процессоров. Вид меню Edit среды Visual C++ показан на рис. 2.3.

### Undo.

Команда Undo (отменить)позволяет отменить последнее изменение, которое было внесено в процессе редактирования. Вы можете также выполнить эту команду с помощью панели инструментов. На кнопке Undo изображена стрелочка влево; эта кнопка расположена седьмой слева.

### Redo.

Команда Redo (восстановить)позволяет отменить действие последней команды Undo. Пользуйтесь этой командой, чтобы вернуть те изменения,которые вы по ошибке отменили. Команда Redo доступна также посредством панели инструментов. Это восьмая кнопка слева.

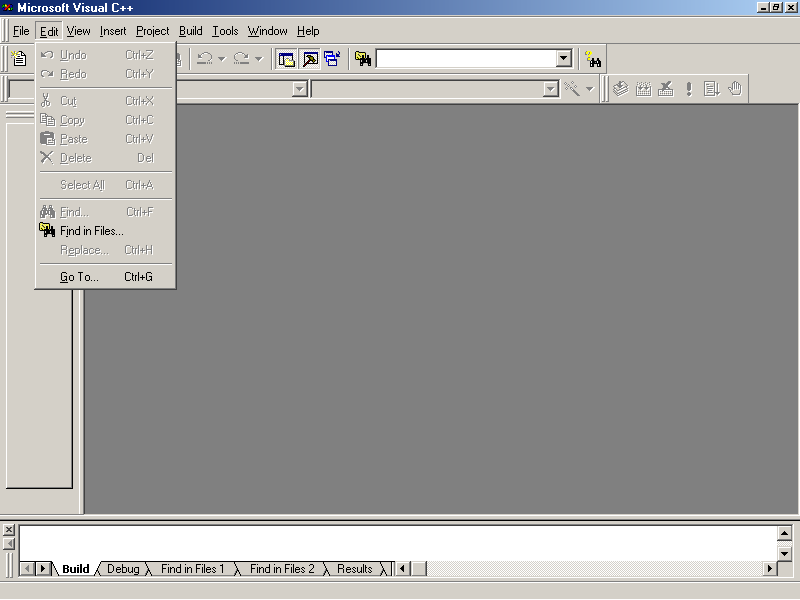


Рис. 2.3. Меню Edit среды Visual C++

### Cut.

Команда Cut (вырезать)переносит в буфер обмена текст, выделенный в активном окне. При этом выделенный текст будет удален из окна. Для того чтобы выделить текст, поместите указатель мыши на первый из символов выделяемого текста и, удерживая левую кнопку нажатой, переместите мышь вправо/вниз до конца выделяемого участка. После этого текст, который вы выделили, будет отображаться инверсным цветом (негативно).

Команда Cut часто используется в паре с командой Paste (вставить) для переноса фрагмента текста с одного места на другое. Когда текст помещается в буфер обмена, все предыдущее содержимое буфера исчезает.

Команду Cut также можно выполнить из панели инструментов. На соответствующей кнопке изображены ножницы; это четвертая кнопка слева.

### Сору.

Как и Cut, команда Сору (копировать)помещает выделенный текст в буфер обмена, но, в отличие от Cut, Copy оставляет исходный текст на месте. Эта команда пригодится при копировании сложных участков текста программы или комментариев, которые требуются в нескольких местах программы.

### Paste.

Команда Paste (вставить)используется, чтобы вставить содержимое буфера обмена туда, где в данный момент находится курсор. Вставлять из буфера обмена можно только то, что было ранее помещено туда командой Cut или Сору.

Команду Paste можно выполнить, используя панель инструментов. Кнопка для этой команды имеет изображение страницы буфера обмена. Это шестая кнопка слева.

### Delete.

Команда Delete (удалить)уничтожает выделенный текст, не помещая его в буфер обмена. Для того чтобы выделить текст, поместите указатель мыши. На первый из символов выделяемого текста и, удерживая левую кнопку нажатой, переместите мышь вправо/вниз до конца выделяемого участка. После этого текст, который вы выделили, будет отображаться инверсным цветом (негативно).

Несмотря на то, что удаляемый текст не переносится в буфер, вы можете отменить удаление, выбрав команду Edit|Undo (правка|отменить).

### Select All.

Команда Select All (выделить все)используется, чтобы выделить все содержимое активного окна для последующей вырезки, копирования или удаления.

### Find...

Команда Find... (поиск)работает аналогично команде поиска в текстовом процессоре. Поскольку в языках С и C++ различаются прописные и строчные буквы, команду Find... можно настроить для поиска с учетом регистра, без учета регистра, а также поиска только полного слова. Команда Find также позволяет указать направление поиска (вперед или назад от текущей позиции курсора).

Одна из полезных нетривиальных возможностей команды Find, которой обычно не бывает в текстовых процессорах, включается флажком опции Regular Expressions (поиск по шаблону). В табл. 2.2 приведены шаблоны, которые можно использовать в строке Find What (что найти).

Таблица 2.2.

|  |  |
| --- | --- |
| **Шаблон** | **Значение** |
| **\*** | Задает любую последовательность символов  Пример: Data \* 1  Соответствует Data1, Dataln1,Dataout1 |
| **.** | Задает один (любой) символ  Пример: Data.  Соответствует Data1, Data 2, но не Dataln1. |
| **^** | Отмечает начало строки  Пример: ^ do  Соответствует любой строке, начинающейся с do |
| **+** | Задает любое число символов перед искомым словом  Пример: +value  Соответствует: i\_value, fvalue, ing\_value. |
| **$** | Отмечает конец строки.  Пример: some\_var\_n;$  Соответствует любой строке, заканчивающейся символами some\_var\_n; |
| **[]** | Задает определенный набор символов  Пример: Data [A…Z]  Соответствует DataA, но не Data1  Пример: Data [1248]  Соответствует Data2, Но не Data4 |
| **\** | Задает два определенных набора символов  Пример: Data [A…Z\0…9]  Соответствует DataAi1, но не DataDo3 |
| **\{\}** | Задает между скобками последовательность символов  Пример: \{no\}\*\_answer  Соответствует \_answer, no \_answer, nono\_answer, nonono\_ answer |

### Replace...

Команда Replace... (заменить)вызывает окно диалога Replace, которое позволяет найти текст по образцу и заменить на заданный. Введите образец, который следует найти, затем укажите текст, на который его следует заменить, и наконец, выберите один из типов соответствия образцу. Можно задать соответствие с учетом регистра, без учета регистра, только полное слово или регулярное выражение.

Будьте осторожны с опцией Replace All (заменить все), поскольку она может привести к нежелательным результатам. При пользовании командой замены следует помнить две вещи: во-первых, сохраните файл перед выполнением замены; во-вторых, если что-то не так, всегда можно воспользоваться командой Undo.

### Go To...

Среда Visual C++ снова приходит к вам на помощь. Поскольку приложение Windows обычно строится из десятков файлов, иногда не так уж просто найти место, где находится описание интересующего вас объекта. Это сделает для вас команда Go To... (перейти к...). Чтобы найти строку с описанием константы, переменной или функции, установите курсор на имя объекта (или непосредственно слева от него) и выполните эту команду. Интегрированная среда автоматически проследит, где находится описание данного имени.

### InfoViewer Bookmarks...

На панели InfoViewer показано оглавление электронных книг. Вы можете вызвать любой раздел из этой иерархии.

### Bookmark.

Команда Bookmark (закладка)устанавливает или удаляет закладку. Закладки используются для того, чтобы отмечать некоторые строки в тексте Программы, к которым придется обратиться позже. Закладки можно установить как этой командой, так и при помощи параметра Set Bookmark (установить закладку) команды Find...

### Breakpoints...

Команда Breakpoints... (точки останова)открывает окно диалога Breakpoints, позволяющее добавлять, удалять, выключать отдельные точки останова, а также удалить все точки сразу. Точки останова затем могут быть использованы командой Go (запустить).

Самый простой способ установить точку останова — нажать на кнопку Breakpoint (если панель инструментов находится на экране). Это вторая кнопка справа; на ней нарисована рука, подающая сигнал "стоп". Кнопка Breakpoint — переключатель: если строка, где стоит курсор, не содержит точки останова, в нее будет помещена новая точка. Если же в строке уже стояла точка останова, она будет удалена.

Таким способом можно установить любое количество точек останова. Теперь каждый раз, когда выполняется команда Go, программа будет выполняться от текущей строки до следующей точки останова.

### Properties...

Команда Properties... (свойства)вызывает окно диалога, посредством которого задаются характеристики текущего окна. К примеру, файл в активном окне можно пометить "только для чтения" или указать, что он может содержать конструкции С, но не C++. Команда Read Only (только для чтения) позволяет переключать в текущем окне параметр "только для чтения". Эта команда изменяет соответствующий атрибут файла. Если файл в данный момент доступен для записи, он будет сделан доступным только для чтения. Если файл был ранее объявлен доступным только для чтения, это ограничение будет снято, и вы сможете сохранить сделанные в нем изменения. Используйте эту команду, если вы хотите сделать файл неизменяемым. Этот режим не позволит внести в файл какие-либо изменения. Такая возможность, в частности, полезна при просмотре файлов, необходимых компилятору, например, стандартных заголовочных файлов. Если бы вы случайно изменили один из них, это вполне могло бы привести к сбоям в работе компилятора.

## Меню View.

Меню View (просмотр), показанное на рис. 2.4, обеспечивает доступ к командам, применяемым для отображения текущего проекта в разных видах.Сюда входят средства просмотра файлов справки, электронных книг, а также другие средства, облегчающие отладку приложений.

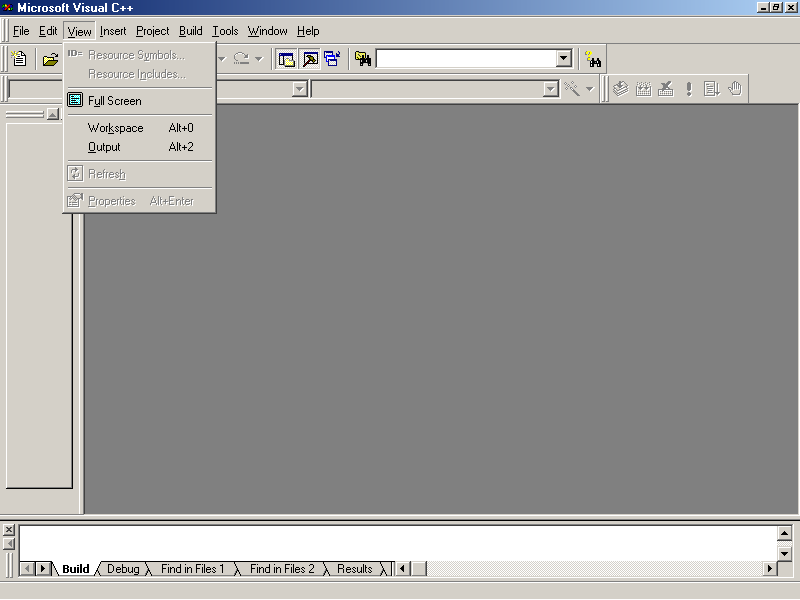


Рис. 2.4. Меню View

### ClassWizard...

Команда ClassWizard... (мастер классов)вызывает средство высокоуровневого программирования, которое позволяет объявлять новые классы на основе компонент Microsoft Foundation Class (MFC), либо добавлять новые методы обработки сообщений в существующий объект, построенный на основе MFC.

### Resource Symbols...

Вы можете использовать команду Resource Symbols... (символы ресурсов)для отображения списка символов, используемых в данном файле ресурсов.

### Resource Includes...

Обычно Microsoft Developer Studio (мастерская разработчика) хранит все ресурсы в файле с расширением RC, а определения символов — в файле RESOURCE.H. При помощикоманды Resource Includes... (включаемые файлы ресурсов) вы можете изменить распределение ресурсов в файлах.

### Full Screen.

При помощи команды Full Screen (полный экран)вы можете развернуть текстовый редактор (или другие редакторы ресурсов) на полный экран. При этом на панели инструментов появится кнопка с маленьким изображением экрана компьютера. Нажав на нее, вы сможете переключиться из полноэкранного в обычный режим и обратно.

### Toolbars...

Команда Toolbars... (панели инструментов)вызывает появление окна диалога Toolbars, позволяющее выбрать панели инструментов, которые должны отображаться в среде Visual C++.

Если вы любите клавиатуру и предпочитаете работать только посредством горячих клавиш, либо хотите освободить побольше места для редактирования, вы можете убрать панели инструментов, сняв соответствующие флажки В окне диалога. Панели инструментов, расположенные под строкой меню, исчезнут. Если же вы потом решите, что вам нравится использовать панели Инструментов, вы можете снова включить их посредством того же окна диалога. В окне названия включенных панелей помечены слева галочкой.

### InfoViewer Query Results.

Команда InfoViewer Query Results (результаты запросов InfoViewer)показывает список предыдущих запросов на поиск информации в справочной системе.

### InfoViewer History List.

Команда InfoViewer History List (хронологический список InfoViewer)показывает список разделов справки, к которым вы недавно обращались.

### Project Workspace.

Команда Project Workspace (рабочее пространство проекта)выдает окно с закладками, соответствующими активным окнам текущего проекта.

### Info Viewer Topic.

Команда InfoViewer Topic (раздел InfoViewer)показывает содержание текущего раздела справочной системы.

### Output.

Команда Output (вывод)активизирует окно Output для данного проекта.

### Watch.

Команда Watch (наблюдение)отображает окно отладчика Watch. Это окно используется для наблюдения за значениями переменных по ходу выполнения программы.

### Variables.

Команда Variables (переменные)открывает окно Variables, позволяющее увидеть активные в данный момент переменные, локальные переменные, или указатель this.

### Registers.

Команда Registersвызывает окно регистров. Это окно позволяет считывать (но не модифицировать) состояние процессора, в том числе содержимое регистров и состояние флагов.

### Memory.

Команда Memory (память)открывает окно, отображающее содержимое памяти с заданного адреса. Окно дает возможность просматривать состояние памяти в пределах адресного пространства программы.

### Call Stack.

Команда Call Stack (стек вызовов)вызывает соответствующее окно диалога. В нем показана последовательность вложенных вызовов функций до уровня, на котором находится текущая строка текста. Функции перечислены в стековом порядке: последняя вызванная функция находится сверху, первая — в самом низу списка.

Если установить флажок опции Show Function Parameters (показать параметры функций), отладчик будет показывать адреса параметров, передаваемых функциям.

### Disassembly.

Команда Disassembly (дизассемблер)отображает окно с ассемблерным кодом, который был сгенерирован из ваших исходных текстов.

## Меню Insert.

Посредством меню Insert (вставить)вы сможете вставить файлы (Files...), ресурсы (Resources), копии ресурсов (Resource Copies), файлы в проект (Files into a Project), проекты в проект (Project) и компоненты рабочего пространства (Components...). Меню Insert показано на рис. 2.5.

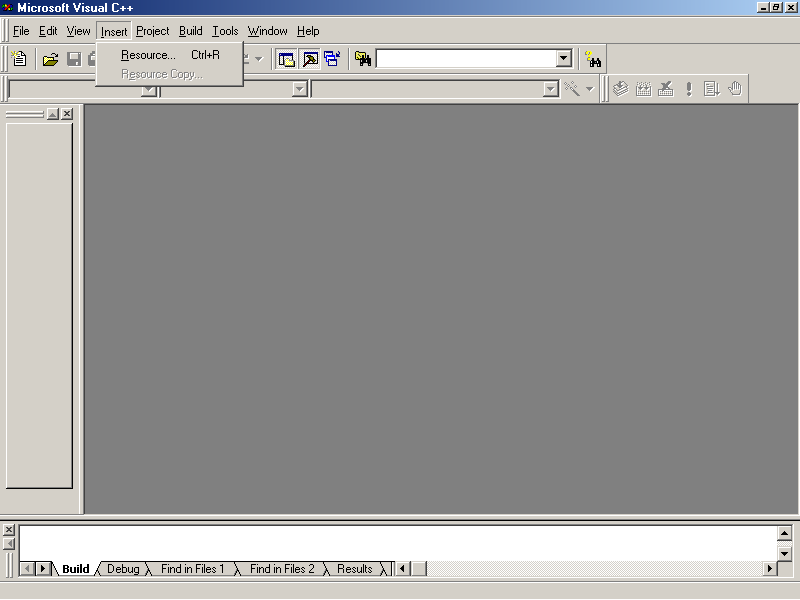


Рис. 2.5. Меню Insert Visual C++

## Меню Build.

Меню Build (сборка), изображенное на рис. 2.6, содержит много пунктов, необходимых для генерации исполняемого файла приложения (\*.ЕХЕ).

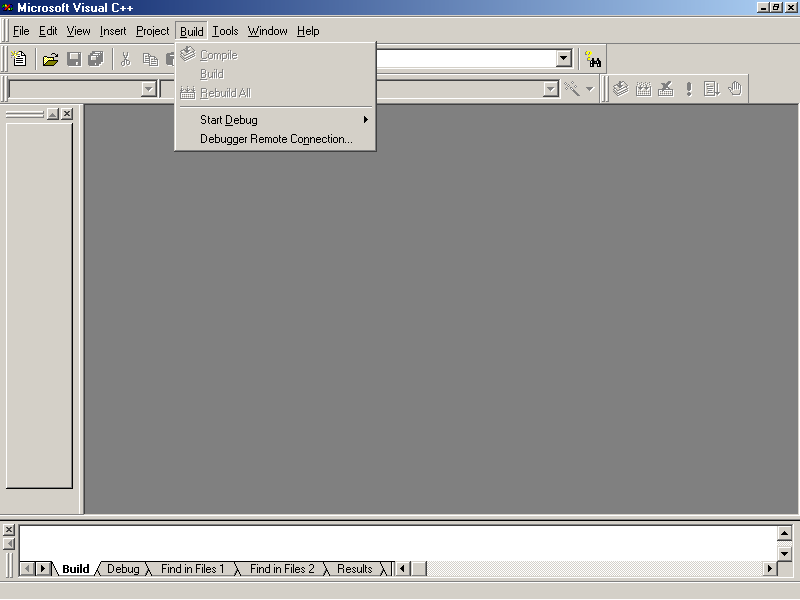


Рис. 2.6. Меню Build Visual C++

### Compile.

Команда Compile (компиляция)заставляет среду компилировать текст в текущем окне. Компиляция — важный этап разработки, поскольку именно здесь вы узнаете, содержит ли данный файл синтаксические ошибки. По этой причине существует возможность компилировать заголовочные файлы (\*.Н), несмотря на то, что они не могут исполняться. Если при компиляции обнаружатся ошибки (или предупреждения о возможных ошибках), сообщения о них будут помещены в окно Output.

### Build.

Обычно программы на C/C++ состоят из множества файлов. Эти файлы могут поставляться с компилятором, с операционной системой, создаваться программистом или даже приобретаться у сторонних производителей. Картина еще более усложняется, если файлы, составляющие проект, создаются несколькими группами программистов. Поскольку файлов очень много, а компиляция может занимать много времени, команда Build (сборка) становится чрезвычайно полезной. В процессе сборки анализируются все файлы проекта и затем компилируются и компонуются лишь те из них, которые были изменены после создания исполняемого файла проекта.

Есть один вопрос, который надо решить перед вызовом команды Build: включать ли в исполняемый файл отладочную информацию (Debug Mode — отладочный вариант) или не включать (Release Mode — окончательный ва-риант). Когда ваша программа готова, вам следует выбрать сборку без отладочной информации, поскольку включение этой информации приведет к ненужному увеличению размера файла.

Если при сборке обнаружатся ошибки (или предупреждения о возможных ошибках), сообщения о них будут помещены в окно Output.

Если включена панель инструментов, вы можете выполнить сборку, нажав шестую кнопку справа. На кнопке нарисовано что-то вроде ведра с двумя темными стрелочками, направленными вниз.

### Rebuild All.

Единственная разницамежду командами Build и Rebuild All (полная сборка)состоит в том, что Rebuild All не обращает внимания на даты изменения файлов и терпеливо компилирует и компонует все.

Представьте себе такую ситуацию. Ваша фирма ради экономии решила обойтись без работников технического обслуживания. По этой причине, да еще из-за перехода на летнее время, простоев и так далее, часы на всех компьютерах в сети показывают разное время. Может случиться, что измененный сегодня файл будет иметь вчерашнюю дату! Если выполнить команду Build, то эти изменения не будут учтены при построении исполняемого файла. Напротив, если воспользоваться командой Rebuild All, можно избежать проверок даты/времени и создать исполняемый файл, действительно отражающий текущее состояние всех файлов проекта.

Если при сборке обнаружатся ошибки (или предупреждения о возможных ошибках), сообщения о них будут помещены в окно Output. Для перемещения по списку ошибок используйте команды Next Error (следующая ошибка) и Previous Error (предыдущая ошибка).

Если панель инструментов включена, вы можете выполнить команду Rebuild All, нажав пятую справа кнопку. На ней изображено ведро с тремя светлыми стрелками вниз.

### Batch Build...

Команда Batch Build... (пакетная сборка)работает аналогично обычной сборке, но может создать в одном проекте сразу несколько целевых файлов.

### Stop Build.

Используйте команду Stop Build (прервать сборку),если вы хотите остановить процесс сборки. Одна из возможных причин такой необходимости — ситуация, когда вы внезапно вспомнили, что не включили в проект один из необходимых файлов.

### Update All Dependencies.

Команда Update All Dependencies (обновить список зависимостей)может быть использована, если вы хотите, чтобы система заново построила диски включаемых файлов для каждого из исходных файлов проекта. Эта команда автоматически добавляет в проект файл, находящийся в активном окне. Используйте эту команду после добавления в текущий файл кодовых включаемых файлов (например, LIMITS.H).

### Debug.

Команда Debug (отладка)запускает интегрированный отладчик.

### Execute.

Команда Execute (выполнить)используется для запуска вашей программы.В зависимости от типа целевого файла Visual C++ автоматически вызовет среду MS-DOS, Windows 95 или Windows NT для тестирования вашей программы.

### Settings...

Команда Settings... (параметры)используется для выбора конфигурации проекта. Вы можете также выбрать несколько конфигураций и задать общие для них настройки. Выбирайте нужную страницу окна диалога при помощи закладок. На страницах C/C++ и Link (компоновка) вы можете задать настройки из различных категорий, пользуясь списком Category (категории) в верхней части.

### Configurations...

Команда Configurations...(конфигурации)служит для управления конфигурациями проекта. Когда вы создаете новый проект, Developer Studio устанавливает стандартные параметры и для отладочной, и для окончательной конфигурации. Отладочная конфигурация содержит всю отладочную информацию, которая может быть использована встроенным отладчиком или любым другим, распознающим формат отладочной информации фирмы Microsoft. Кроме того, Developer Studio отключает любую оптимизацию отладочной версии, поскольку оптимизация иногда усложняет процесс отладки. Окончательная конфигурация не содержит какой-либо отладочной информации и использует оптимизацию, установленную вами после создания проекта. В зависимости от настроек среды и параметров, заданных при создании проекта, у вас могут быть и другие конфигурации с другими настройками. Вы также можете сами создать дополнительные конфигурации. Каждая конфигурация проекта определяет также каталоги, куда будут помещены файлы с промежуточными и окончательными результатами компиляции и компоновки.

### Subprojects...

Microsoft Developer Studio позволяет вам включать в проект другие проекты. Доступ к ним осуществляется при помощикоманды Subprojects... (подпроекты).

### Set Default Configuration...

В большинстве случаев достаточно настроить конфигурацию для всего проекта в целом, но при необходимости вы можете задать для конкретных файлов проекта другие значения каких-либо параметров. Таким образом, конфигурация имеет иерархическую структуру: параметр, заданный на уровне проекта, относится ко всем файлам, кроме тех, для которых он переопределен на уровне файла. Например, если при помощи команды Set Default Configuration (установить основную конфигурацию) вы установили параметр Default optimization (стандартная оптимизация), то эта установка будет действовать на все файлы в данной конфигурации. Тем не менее, вы можете установить конкретные настройки оптимизации (или отключить ее вообще) для каких-либо файлов в отдельности. Значения параметров, заданные на уровне файла, будут использованы вместо тех, которые заданы на уровне проекта. Некоторые виды параметров (например, параметры компоновки) можно задавать только на уровне проекта.

Вы можете задавать конфигурационные параметры на уровне проекта или файла. Значения параметров, заданные на уровне проекта, используются для каждого файла, если они не переопределены на уровне файла.

Установки на уровне файла относятся только к действиям, выполняемым с файлом, например, компиляции. Значения параметров, заданные на уровне файла, относятся только к этому файлу и используются вместо значений, заданных на уровне проекта.

## Меню Tools.

Меню Tools (инструменты), показанное на рис. 2.7, обеспечивает доступ к множеству полезных вспомогательных средств интегрированной среды. Эти средства призваны облегчить, насколько это возможно, процесс разработки и модификации различных объектно-ориентированных приложений Windows.

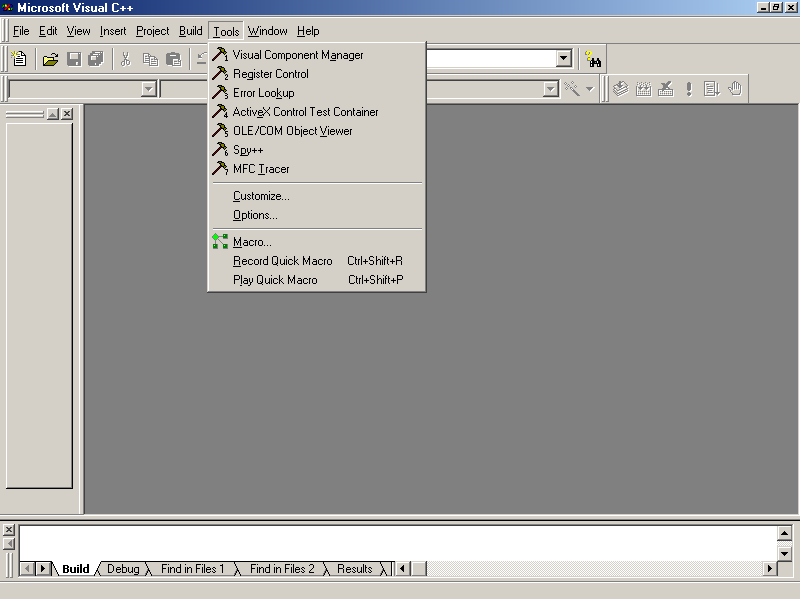


Рис.2.7.Меню Tools

### Browse...

Команда Browse... (найти)выдаст хронологический список описаний объектов или ссылок на них, которые вы искали ранее. Список устроен наподобие стека, то есть при добавлении новый элемент становится первым в списке.

### Close Browse Info File.

Команда Close Browse Info File (закрыть файл информации броузера)закрывает файл Browse Info (информация броузера).

### OLE Control Test Container.

Эта команда вызывает утилиту OLE Control Test Container (тестовый контейнер для управляющих элементов OLE). Это приложение, которое полностью поддерживает управляющие элементы OLE и позволяет встраивать их в свои окна и окна диалога. Управляющий элемент OLE представляет собой многократно используемый программный модуль, который можно использовать в нескольких программах. Управляющие элементы дают пользователю возможность обращаться к базам данных, отслеживать данные и выполнять различные команды прикладных программ.

### OLE Object View.

Команда OLE Object View (показать объект OLE)вызывает OLE Object Viewer (средство просмотра объектов OLE). OLE Object Viewer — это вспомогательная программа для того, чтобы помочь разработчикам OLE-приложений лучше понять, что происходит в их системах. Это также мощный отладочный инструмент, который позволяет проверить, что ваши объекты и интерфейсы ведут себя именно так, как планировалось.

## Меню Window.

Меню Window, изображенное на рис. 2.8, позволяет управлять отображением различных окон, используемых в процессе разработки приложения. Это меню также позволяет устанавливать фокус ввода, указывая, какое окно сделать активным.

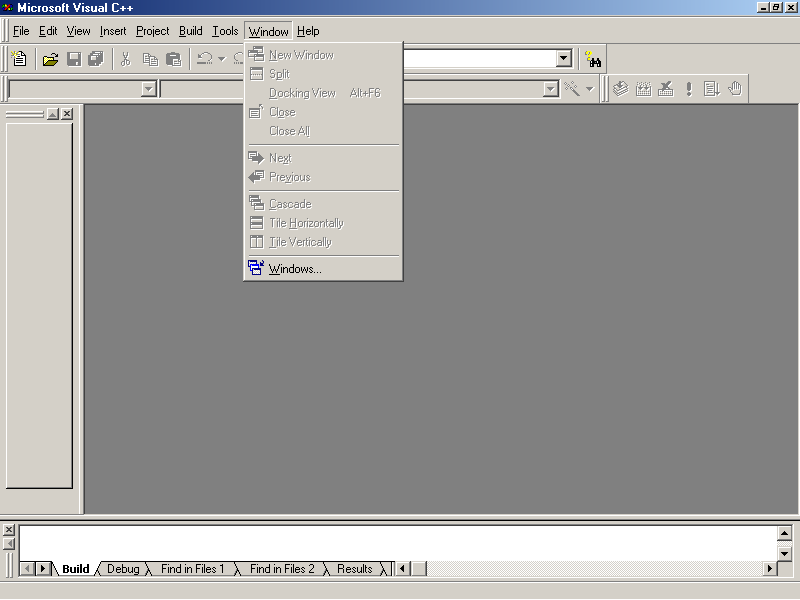


Рис.2.8. Меню Window

### New Window.

Команда New Window (новое окно)дает еще одну возможность создать в текущем проекте новое окно.

### Split.

Команда Split (разделить)делит текущее окно на две части.

### Hide.

Команда Hide (спрятать)делает текущее окно скрытым.

### Cascade.

Команда Cascade (каскад)располагает все открытые на экране окна одно за другим со сдвигом вправо и вниз, наподобие колоды карт. Такое расположение позволяет сразу видеть количество окон и имена файлов, связанных с ними. Недостаток состоит в том, что вам видно содержимое лишь самого верхнего окна.

### Tile Horizontally, Tile Vertically.

Команды Tile Horizontally (сверху вниз) и Tile Vertically (справа налево)заставляют среду поровну разделить экран между всеми окнами. Преимущество такого расположения состоит в том, что вы можете одновременно видеть содержимое всех окон. Недостатки проявляются при большом количестве окон: тогда каждому из них достается место размером с почтовую марку.

Команды Tile Horizontally и Tile Vertically располагают окна по вертикали и горизонтали соответственно.

### Close All...

Команда Close All... (закрыть все)закрывает все открытые окна.

### Windows.

Команда Windows (окна)открывает список всех имеющихся в среде окон, позволяя быстро найти нужное.

## Меню Help.

Возможности такой разнообразной, насыщенной и интеллектуальной среды, как Visual C++, могут остаться невостребованными без последнего, но самого важного меню Help (справка). Пытаетесь ли вы разобраться в конструкции языка C/C++, узнать о средстве из среды Visual C++ или научиться использовать их совместно, вам помогут команды, доступные посредством меню Help.

Чтобы ускорить процесс поиска в информационной базе данных, в меню эта база подразделяется на несколько категорий. Каждая команда вызывает стандартное окно диалога справочной системы, позволяющее задать шаблоны поиска, перемещаться вперед и назад по вложенным окнам справки и выбирать способ отображения справочной информации.