

Б. Пахомов



# C/C++ И MS Visual C++ 2008 для начинающих

Основные элементы языков C/C++

Визуальная среда программирования

Создание основных типов приложений

Работа с наборами данных

+ DVD  
Дистрибутив  
Microsoft  
Visual Studio 2008  
Express Edition

  
Microsoft  
**Visual Studio**  
Express Editions

**Борис Пахомов**

**C/C++**

**И**

**MS Visual**

**C++ 2008**

**для начинающих**

Санкт-Петербург

«БХВ-Петербург»

2008

УДК 681.3.068+800.92С/С++  
ББК 32.973.26-018.1  
П22

## **Пахомов Б. И.**

П22 С/С++ и MS Visual С++ 2008 для начинающих. — СПб.: БХВ-Петербург, 2008. — 624 с.: ил. + Дистрибутив (на DVD)

ISBN 978-5-9775-0267-2

Книга является руководством для начинающих по разработке приложений в среде Microsoft Visual С++ 2008 Express Edition. Рассмотрены основные элементы языков программирования С/С++ и примеры создания простейших классов и программ. Изложены принципы визуального проектирования и событийного программирования. На конкретных примерах показаны основные возможности визуальной среды разработки Visual С++ 2008 Express Edition, назначение базовых компонентов и процесс разработки различных типов консольных и Windows-приложений. На компакт-диске размещен дистрибутив пакета Microsoft Visual Studio 2008 Express Edition, содержащий Visual С++ 2008 Express Edition и другие компоненты пакета.

*Для начинающих программистов*

УДК 681.3.068+800.92С/С++  
ББК 32.973.26-018.1



*"Microsoft, Visual Basic, Visual C#, Visual С++, Visual J#, Visual Web Developer, Visual Studio и логотип Visual Studio представляют собой торговые марки Microsoft Corporation, зарегистрированные в США и/или других странах".*

### **Группа подготовки издания:**

Главный редактор	<i>Екатерина Кондукова</i>
Зам. главного редактора	<i>Игорь Шишигин</i>
Зав. редакцией	<i>Григорий Добин</i>
Редактор	<i>Римма Смоляк</i>
Компьютерная верстка	<i>Натальи Смирновой</i>
Корректор	<i>Зинаида Дмитриева</i>
Дизайн обложки	<i>Елены Беляевой</i>
Зав. производством	<i>Николай Тверских</i>

Лицензия ИД № 02429 от 24.07.00. Подписано в печать 30.04.08.

Формат 70×100<sup>1</sup>/<sub>16</sub>. Печать офсетная. Усл. печ. л. 50,31.

Тираж 3000 экз. Заказ №

"БХВ-Петербург", 194354, Санкт-Петербург, ул. Есенина, 5Б.

Санитарно-эпидемиологическое заключение на продукцию № 77.99.60.953.Д.002108.02.07 от 28.02.2007 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 978-5-9775-0267-2

© Пахомов Б. И., 2008

© Оформление, издательство "БХВ-Петербург", 2008

# Оглавление

<b>ВВЕДЕНИЕ.....</b>	<b>1</b>
<b>ГЛАВА 1. ТИПЫ ДАННЫХ, ПРОСТЫЕ ПЕРЕМЕННЫЕ И ОСНОВНЫЕ ОПЕРАТОРЫ ЦИКЛА .....</b>	<b>5</b>
Структура рабочего стола среды программирования .....	5
Главное окно .....	6
Структура программ в VC++ .....	8
Переход к созданию консольного приложения .....	9
Создание простейшего консольного приложения .....	17
Программа с оператором <i>while</i> .....	24
Имена и типы переменных.....	26
Оператор <i>While</i> .....	27
Оператор <i>for</i> .....	30
Символические константы.....	31
<b>ГЛАВА 2. ПРОГРАММЫ ДЛЯ РАБОТЫ С СИМВОЛЬНЫМИ ДАННЫМИ .....</b>	<b>33</b>
Программа копирования символьного файла. Вариант 1 .....	35
Программа копирования символьного файла. Вариант 2.....	38
Подсчет символов в файле. Вариант 1 .....	39
Подсчет символов в файле. Вариант 2 .....	42
Подсчет количества строк в файле .....	45
Подсчет количества слов в файле .....	46
<b>ГЛАВА 3. РАБОТА С МАССИВАМИ ДАННЫХ.....</b>	<b>51</b>
Одномерные массивы.....	51
Многомерные массивы.....	55
<b>ГЛАВА 4. СОЗДАНИЕ И ИСПОЛЬЗОВАНИЕ ФУНКЦИЙ.....</b>	<b>57</b>
Создание некоторых функций.....	59
Ввод строки с клавиатуры.....	60

Функция выделения подстроки из строки.....	63
Функция копирования строки в строку .....	65
Головная программа для проверки функций <i>getline()</i> , <i>substr()</i> , <i>copy()</i> .....	65
Внешние и внутренние переменные .....	68
Область действия переменных .....	72
Как создать свой внешний файл.....	72
Атрибут <i>static</i> .....	73
Рекурсивные функции .....	76
<b>ГЛАВА 5. ФУНКЦИИ ДЛЯ РАБОТЫ С СИМВОЛЬНЫМИ СТРОКАМИ.....</b>	<b>79</b>
Основные стандартные строковые функции.....	79
Пример программы проверки функций.....	81
<b>ГЛАВА 6. ДОПОЛНИТЕЛЬНЫЕ СВЕДЕНИЯ О ТИПАХ ДАННЫХ, ОПЕРАЦИЯХ, ВЫРАЖЕНИЯХ И ЭЛЕМЕНТАХ УПРАВЛЕНИЯ .....</b>	<b>87</b>
Новые типы переменных .....	87
Константы.....	91
Новые операции .....	92
Преобразование типов данных.....	94
Побитовые логические операции .....	95
Операции и выражения присваивания.....	96
Условное выражение .....	99
Операторы и блоки .....	99
Конструкция <i>if-else</i> .....	100
Конструкция <i>else-if</i> .....	100
Переключатель <i>switch</i> .....	105
Уточнение по работе оператора <i>for</i> .....	109
Оператор <i>continue</i> .....	110
Оператор <i>goto</i> и метки.....	110
<b>ГЛАВА 7. РАБОТА С УКАЗАТЕЛЯМИ И СТРУКТУРАМИ ДАННЫХ .....</b>	<b>111</b>
Указатель .....	111
Указатели и массивы .....	116
Операции над указателями.....	118
Указатели и аргументы функций.....	118
Указатели символов и функций.....	120
Передача в качестве аргумента функции массивов размерности больше единицы.....	125
Массивы указателей .....	125
Указатели на функции .....	126

Структуры.....	129
Объявление структур.....	129
Обращение к элементам структур.....	131
Структуры и функции.....	134
Программы со структурами.....	134
Рекурсия в структурах.....	144
Битовые поля в структурах.....	151
Категории памяти.....	152
<b>ГЛАВА 8. КЛАССЫ В C++ .....</b>	<b>155</b>
Объектно-ориентированное программирование.....	155
Классы.....	157
Принципы построения классов.....	158
Примеры создания классов.....	162
Конструкторы и деструкторы класса.....	169
Конструктор класса.....	169
Деструктор класса.....	172
Классы, структуры и массивы в среде CLR.....	173
Классы и структуры.....	173
Массивы.....	175
<b>ГЛАВА 9. ВВОД И ВЫВОД В ЯЗЫКАХ C И C++.....</b>	<b>181</b>
Ввод и вывод в C.....	181
Ввод/вывод файлов.....	181
Стандартный ввод/вывод.....	189
Ввод/вывод в C++.....	208
Общие положения.....	208
Ввод/вывод с использованием разных классов.....	209
Стандартный ввод/вывод в C++.....	222
<b>ГЛАВА 10. ПРОДОЛЖЕНИЕ ИЗУЧЕНИЯ СРЕДЫ VISUAL C++ .....</b>	<b>231</b>
Создание проекта.....	233
Некоторые файлы проекта.....	237
Окно сведений об объекте.....	240
Вкладка <i>Events</i> .....	241
Вкладка <i>Property Pages</i> .....	243
Управление окнами документов.....	243
Работа с окном сведений об объекте.....	245
Редактор кода, h-модуль и режим дизайна (проектирования).....	247

Контекстное меню редактора кода.....	248
Суфлер кода (подсказчик).....	249
Настройка редактора кода.....	250
Начало редактирования кода программного модуля .....	253
Компоненты среды программирования VC++.....	254
Класс <i>Form</i> .....	254
Помещение компонента в форму .....	256
Другие действия с дизайнером форм .....	256
Контекстное меню формы.....	257
Добавление новых форм к проекту .....	258
Организация работы с множеством форм .....	260
Свойства формы.....	260
События формы.....	275
Некоторые методы формы .....	276
Рисование графиков в форме.....	279

## ГЛАВА 11. КОМПОНЕНТЫ, СОЗДАЮЩИЕ ИНТЕРФЕЙС

### МЕЖДУ ПОЛЬЗОВАТЕЛЕМ И ПРИЛОЖЕНИЕМ.....285

Пространство имен <i>System</i> .....	286
Работа с переменными некоторых типов .....	287
Компонент <i>Button</i> .....	291
Свойства <i>Button</i> .....	292
События <i>Button</i> .....	298
Методы <i>Button</i> .....	299
Компонент <i>Panel</i> .....	299
Некоторые свойства <i>Panel</i> .....	300
Некоторые события <i>Panel</i> .....	300
Компонент <i>Label</i> .....	302
Некоторые свойства <i>Label</i> .....	303
События <i>Label</i> .....	304
Компонент <i>TextBox</i> .....	305
Некоторые свойства <i>TextBox</i> .....	305
События <i>TextBox</i> .....	309
Некоторые методы <i>TextBox</i> .....	311
Компонент <i>MenuStrip</i> .....	312
Некоторые свойства опций <i>MenuStrip</i> .....	322
События <i>MenuStrip</i> .....	324
Компонент <i>ContextMenuStrip</i> .....	324
Компонент <i>ListView</i> .....	325
Некоторые свойства <i>ListView</i> .....	327
События <i>ListView</i> .....	333

Компонент <i>WebBrowser</i> .....	335
Компонент <i>ListBox</i> .....	340
Как работать с <i>ListBox</i> .....	341
Свойства <i>ListBox</i> .....	341
Как использовать <i>ListBox</i> .....	346
Как формировать список строк.....	347
Компонент <i>ComboBox</i> .....	355
Свойства <i>ComboBox</i> .....	356
События <i>ComboBox</i> .....	358
Некоторые методы <i>ComboBox</i> .....	359
Примеры использования <i>ComboBox</i> .....	361
Компонент <i>MaskedTextBox</i> .....	369
Компонент <i>CheckedListBox</i> .....	374
Пример: домашний телефонный справочник.....	379
Компоненты <i>CheckBox</i> и <i>RadioButton</i> .....	401
Компонент <i>GroupBox</i> .....	405
Компонент <i>LinkLabel</i> .....	406
Компонент <i>PictureBox</i> .....	421
Некоторые свойства компонента <i>PictureBox</i> .....	422
Компонент <i>DateTimePicker</i> .....	426
Форматные строки даты и времени.....	429
Компонент <i>TabControl</i> .....	438
Как задавать страницы.....	439
Некоторые методы <i>TabControl</i> .....	443
Некоторые свойства страницы <i>TabPage</i> .....	444
Задача регистрации пользователя в приложении.....	447
Компонент <i>Timer</i> .....	459
Компонент <i>ProgressBar</i> .....	462
Компонент <i>OpenFileDialog</i> .....	463
Компонент <i>SaveFileDialog</i> .....	471
Компонент <i>ColorDialog</i> .....	480
Компонент <i>FontDialog</i> .....	481
Компонент <i>PrintDialog</i> .....	482
Компонент <i>ToolStrip</i> .....	482
Некоторые свойства <i>ToolStrip</i> .....	484
Использование <i>ToolStrip</i> .....	486

## **ГЛАВА 12. РАБОТА С НАБОРАМИ ДАННЫХ.....487**

Общие сведения о базах данных.....	487
Проектирование баз данных.....	489



Модель базы данных .....	489
Структура проектирования БД .....	490
Идентификация сущностей и атрибутов.....	490
Проектирование таблиц .....	492
Определение неповторяющихся атрибутов .....	493
Набор правил при разработке таблицы .....	494
Выбор индексов .....	496
Язык SQL .....	496
Примеры оператора <i>SELECT</i> .....	498
Наборы данных (компонент <i>DataSet</i> ) .....	499
Общая технология организации формирования набора данных в приложении .....	512
Примеры поиска по первичному ключу .....	517
<b>ГЛАВА 13. УПРАВЛЕНИЕ ИСКЛЮЧИТЕЛЬНЫМИ СИТУАЦИЯМИ.....</b>	<b>523</b>
Операторы <i>try</i> , <i>catch</i> и <i>throw</i> .....	524
Пример 1 .....	525
Пример 2 .....	527
Пример 3 .....	532
Функции, выдающие исключения.....	535
<b>ГЛАВА 14. ПРЕОБРАЗОВАНИЕ МЕЖДУ НЕРЕГУЛИРУЕМЫМИ И РЕГУЛИРУЕМЫМИ (РЕЖИМ CLR) УКАЗАТЕЛЯМИ .....</b>	<b>537</b>
Пример 1. Перевод строки <i>String^</i> в ASCII-строку .....	539
Пример 2. Перевод ASCII-строки в <i>String^</i> -строку .....	540
Пример 3. Преобразование <i>String^</i> -строки в строку <i>wchar_t</i> .....	542
Пример 4. Преобразование строки <i>wchar_t</i> в <i>String^</i> -строку .....	544
Пример 5. Маршаллинг <i>native</i> -структуры .....	546
Пример 6. Работа с массивом элементов <i>native</i> -структуры в <i>managed</i> -функции.....	548
Пример 7. Доступ к символам в классе <i>System::String</i> .....	550
Пример 8. Преобразование <i>char*</i> в массив <i>System::Byte</i> .....	551
Пример 9. Преобразование <i>System::String</i> в <i>wchar_t*</i> или <i>char*</i> .....	552
Пример 10. Преобразование <i>String</i> в <i>string</i> .....	554
Пример 11. Преобразование <i>string</i> -строки в <i>String</i> -строку.....	559
Пример 12. Объявление дескрипторов в <i>native</i> -типах .....	560
Пример 13. Работа с дескриптором в <i>native</i> -функции.....	562

<b>ПРИЛОЖЕНИЯ.....</b>	<b>565</b>
<b>ПРИЛОЖЕНИЕ 1. НЕКОТОРЫЕ СОГЛАШЕНИЯ, ПРИНЯТЫЕ В MICROSOFT .....</b>	<b>567</b>
<b>ПРИЛОЖЕНИЕ 2. ПРЕОБРАЗОВАНИЯ МЕЖДУ РАЗЛИЧНЫМИ ТИПАМИ СТРОК .....</b>	<b>568</b>
Преобразование в тип <i>String</i> .....	568
Преобразование в тип <i>char</i> .....	569
Преобразование в тип <i>wchar_t</i> .....	570
<b>ПРИЛОЖЕНИЕ 3. ОСНОВНЫЕ СТРОКОВЫЕ ТИПЫ ДАННЫХ, ПРИНЯТЫЕ В VISUAL C++ 2008 .....</b>	<b>572</b>
Тип <i>char</i> .....	572
Основные стандартные функции для работы со строками типа <i>char</i> .....	572
Пример программы проверки функций.....	574
Тип <i>wchar_t</i> — расширенные символы Unicode.....	579
Основные стандартные функции для работы со строками типа <i>wchar_t</i> .....	579
Тип <i>string</i> .....	583
Операторы .....	594
Тип <i>String</i> .....	595
<b>ПРИЛОЖЕНИЕ 4. ОПИСАНИЕ ПРИЛАГАЕМОГО ДИСКА.....</b>	<b>599</b>
<b>ПРЕДМЕТНЫЙ УКАЗАТЕЛЬ.....</b>	<b>600</b>



# Введение

Эта книга предназначена для начинающих программистов, поэтому многие вопросы раскрыты не во всей глубине, ибо читатель должен научиться пользоваться продуктом, а не тонуть в подробностях. Того, что дано здесь, достаточно для написания консольных приложений, а также многих приложений типа Windows Form. Автор надеется, что полученные знания явятся ступенью для изучения интересного и полезного (несмотря на его недостатки) программного продукта, каким, несомненно, является MS Visual C++ 2008.

Хотя выхода на создание приложений типа ASP.NET Web Form, требующих использования Web Form-классов, которые не подключены к предлагаемому продукту, в этом облегченном и бесплатно-доступном продукте нет, но для начинающего вполне достаточно изучить язык и Windows-формы, чтобы в дальнейшем, имея определенный багаж знаний, заниматься проблемами Интернета. Кстати, и в этом урезанном варианте некоторые выходы в Интернет имеются (например, через компонент `LinkLabel`).

Но, к сожалению, фирма Microsoft в этом издании своего продукта отключила возможность подключения к базам данных в режиме дизайна приложения, ссылаясь на то, что, якобы, это можно сделать и программно. Однако во всей своей сопроводительной документации фирма ссылается только на возможность подключения к базам данных именно в режиме дизайна приложения, оговорив как бы между прочим, что это подключение можно делать и программно. Но как это делать, фирма скромно умалчивает. В Интернете можно найти массу пособий, указывающих, как это делать программно, однако, во-первых, эти пособия не для начинающих, а во-вторых, ни одно из них не работает сегодня, ибо они многолетней давности, а за это время изменились не только компиляторы, но и библиотеки классов, на которых строились примеры авторов этих пособий. Остается только надеяться, что авторы фирмы Microsoft одумаются и в следующих версиях своего продукта не станут так существенно осложнять жизнь своего пользователя.

Однако все же утешением является оставленная возможность работоспособности программ версии 2005 в новой версии 2008. Поэтому читатели книги могут научиться подключаться к базам данных, изучив *главу 12*, построенную на использовании версии 2005. Читатель может возразить: а к чему выпускать новую книгу, если один из главных моментов (работа с базами дан-

ных) идет по старой книге? Хочу отметить, что данная книга не только уточняет многие моменты, опущенные в прежнем издании, но и дополнена новыми примерами и новыми главами. Поэтому на ее фундаменте начинающему программисту будет значительно легче войти в среду Visual C++, нежели с помощью моей предыдущей книги.

Что касается не вошедших в рассматриваемую среду программирования других типов приложений, то об их сути можно сказать следующее.

- ASP.NET (Active Server Pages — активные серверные страницы) — это унифицированная платформа для разработки Web-приложений, обеспечивающая необходимым сервисом разработчиков таких приложений. Это компилируемая, основанная на NET-технологии среда, в которой можно создавать приложения на любом совместимом с этой технологией языке, включая Visual Basic .NET, C# и JScript .NET.
- XML Web-services — это технология по обеспечению сервисов, которые могут быть доступны через Интернет по запросу с компьютера клиента серверу, имеющему такие сервисы.

XML Web-service — это некий элемент, обладающий определенной функциональностью (например, какая-то программа), доступный через Интернет различным системам, соблюдающим стандарты Интернета (такие как XML и HTTP).

XML Web-service может использоваться самим приложением или выставляться за пределы компьютера через Интернет для многих других приложений (сообщения идут на языке XML). Разработчики XML Web-service могут создавать приложения из различных элементов точно так же, как и другие разработчики используют компоненты при формировании собственных приложений.

Одной из ключевых характеристик XML Web-сервисных приложений является высокая степень абстракции между потребителем и поставщиком сервиса. Использование механизма XML-сообщений, с помощью которого сервис создается, а также осуществление к нему доступа освобождает и клиента, и провайдера (поставщика сервиса) от необходимости обладать такими знаниями, как ввод/вывод и размещение сервиса. XML Web-service в настоящее время становится тем фундаментом, который связывает вместе все компьютерные устройства.

- Windows-service — это программа-приложение, которая запускается в фоновом разделе операционной системы Microsoft Windows NT, Windows 2000, Windows XP или Windows Server 2003. Примером служит

известная программа Telnet, которая управляет удаленным компьютером, или Windows-service времени, который обновляет компьютерные часы.

Windows-service не может запускаться непосредственно из среды Visual C++. Он запускается из специальной среды продукта, называемого Services Control Manager (менеджер управления сервисами). Таким образом, Microsoft Windows-service, известные формально как NT-сервисы, позволяют создавать приложения, запускаемые в своих собственных Windows-сеансах.

Сервисы могут автоматически запускаться в момент загрузки компьютера, могут быть приостановлены и снова запущены, не показывая никакого пользовательского интерфейса. Эти свойства делают такие сервисы идеальными для использования в серверах или где-либо в приложениях в том случае, если не требуется пересекаться с другими пользователями, работающими на том же компьютере.

Создается Windows Service-приложение, которое после разработки и построения надо установить в качестве сервиса с помощью запуска специальной утилиты. Только после этого его можно использовать в среде Services Control Manager, запустив последний (с его помощью можно запускать, приостанавливать, останавливать, перезапускать и конфигурировать созданный сервис).

Что можно сказать об отличиях новой версии (Visual C++ 2008) от версии Visual C++ 2005?

Остановимся на тех из них, которые необходимы начинающему изучение среды. Например, изменения внесены в саму среду разработки приложений IDE (Integrated Development Environment). В частности, по данным авторов разработано более 150 методов для 18 новых или существовавших ранее классов с целью поддержки свойств операционной системы Windows Vista.

Улучшены свойства компилятора, создана специальная библиотека (библиотека маршаллинга), значительно облегчающая процесс преобразования типов данных между так называемыми native- и managed-средами (проще говоря — между старой обычной средой программирования в C++ и новой CLR-средой (Common Language Runtime)). CLR-среда обеспечивает межязыковую совместимость программ, когда они исполняются, а также легко осуществляет преобразование одного типа данного в другой. Добавили авторы и несколько демонстрационных примеров.

### **ВНИМАНИЕ**

Полную информацию о продуктах Visual Studio Express Edition 2008 на русском языке можно получить на сайте [www.microsoft.ru/express](http://www.microsoft.ru/express).



## ГЛАВА 1




# Типы данных, простые переменные и основные операторы цикла

## Структура рабочего стола среды программирования

Цель этой главы — продемонстрировать начальные элементы программирования на языке C. Однако, чтобы построить программу на этом языке, нам надо воспользоваться средой программирования Visual C++ 2008 (ее английская аббревиатура — IDE: Integrated Development Environment), которая содержит в себе средства создания программы, ее компиляции, отладки и запуска на выполнение. В этой связи рассмотрим кратко структуру этой среды, а точнее ее интерфейс, с нами, пользователями. Интерфейс — это аппарат, который позволяет удобно взаимодействовать пользователю со средой.

После установки на своем компьютере среды Visual C++ 2008 вы можете ее загрузить, воспользовавшись командой главного меню **Пуск|Программы** (после установки продукта, его имя автоматически появится в списке команды **Программы**).

Для удобства дальнейшей работы с установленным программным продуктом следует мышью перетянуть его значок  на линейку быстрого запуска программ, которая находится на рабочем столе операционной системы (обычно ее располагают в нижней части стола). Находящийся на этой линейке любой программный продукт запускается одинарным щелчком мыши на значке соответствующего продукта. Итак, загружаем наш продукт Microsoft Visual C++ 2008 Express Edition (для краткости в дальнейшем станем его называть VC++). На экране появится главное окно — рабочий стол, структуру которого мы и рассмотрим.



## Главное окно

Общий вид окна показан на рис. 1.1.

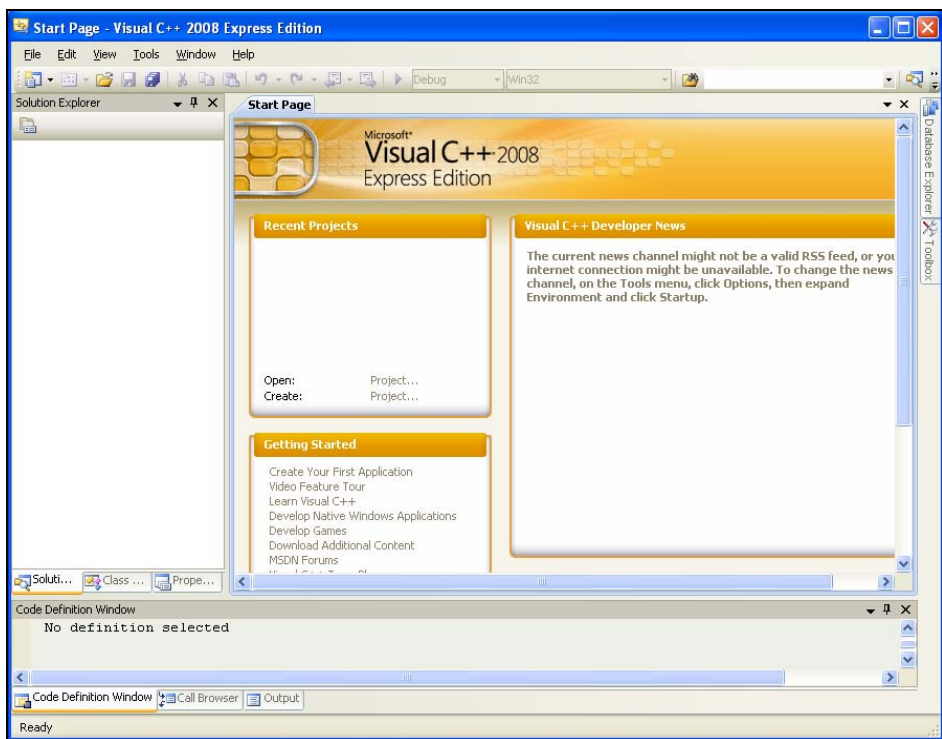


Рис. 1.1. Вид главного окна IDE после загрузки VC++

В верхней части окна расположена строка с командами главного меню среды (команды: **File**, **Edit**,...) — это строка горизонтального меню. При вызове этих команд (их еще называют опциями, т. е. элементами выбора из нескольких значений) открываются так называемые "выпадающие меню" — это вертикальные меню, представляющие собой набор команд, располагающихся на экране сверху вниз. Пример такого меню показан на рис. 1.2.

Во второй строке главного окна, расположенной под строкой главного меню, находятся кнопки быстрого вызова некоторых команд на исполнение. Все эти кнопки имеют всплывающие подсказки (надо привести курсор мыши на кнопку, немного подождать, после чего появится подсказка о том, для чего предназначена данная кнопка).

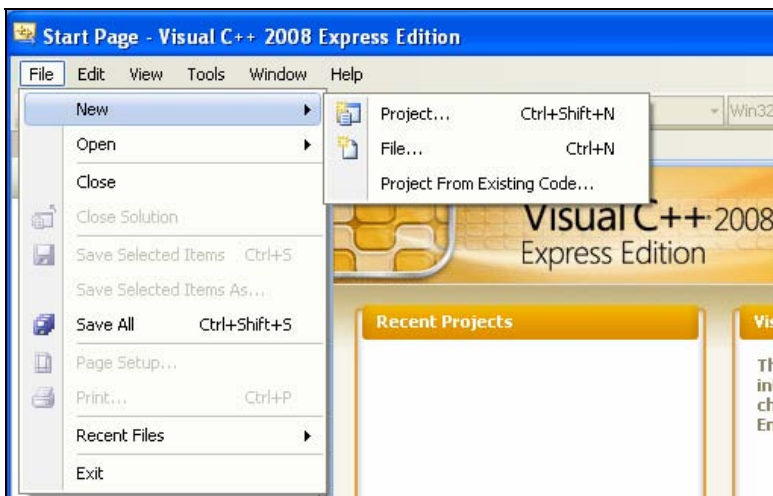


Рис. 1.2. Меню для задания типа формируемого приложения

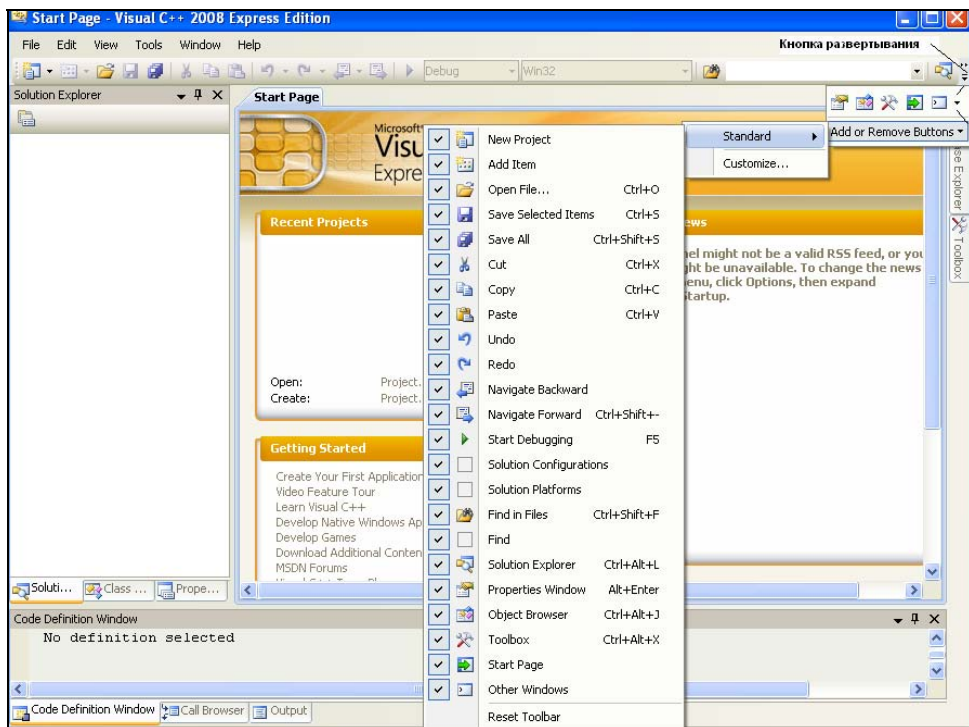


Рис. 1.3. Кнопки быстрого вызова

Рядом с такими кнопками могут быть дополнительные кнопки для раскрытия списка значений основной кнопки. Так как все кнопки не помещаются в отведенное им место на рабочем столе, то они свернуты в небольшие полосы с кнопками их развертывания точно так же, как это выполнено во всем известном Word'e (рис. 1.3).

Вид главного окна, в свою очередь, изменяется при задании типа создаваемого приложения. С этим мы познакомимся, когда начнем создавать приложения.

## Структура программ в VC++

Программы в VC++ называются приложениями (очевидно, приложениями в среде IDE). Мы так и дальше станем их называть. Приложения строятся средой в виде специальных конструкций — проектов, которые выглядят для пользователя как совокупность нескольких файлов.

Программа на языке C — это совокупность функций, т. е. специальных программных образований, отвечающих определенным требованиям. Причем приложение — это главная функция, внутри которой помещаются операторы, реализующие алгоритм приложения. Среди операторов имеются такие, которые служат для вызова других функций, требующихся при реализации алгоритма. Запуск любой программы начинается с запуска *главной функции*, содержащей в себе всю остальную часть программы. Часть функций создается самим программистом, другая часть — библиотечные функции — поставляется пользователю со средой программирования и используется в процессе разработки программ. При изучении C/C++ мы будем пользоваться специальным видом приложений — консольными приложениями, которые формируются на основе заранее заготовленных в среде проектирования шаблонов.

Консольные (т. е. опорные, базовые) приложения — это приложения без графического интерфейса, которые взаимодействуют с пользователем через специальную командную строку или (если они работают в рамках IDE) запускаются специальной командой из главного меню среды. Такие приложения создаются с помощью специального шаблона, доступного из диалогового окна, открывающегося после выполнения команды **File|New Project**.

Шаблон консольного приложения добавляет в создаваемое приложение необходимые элементы (создается заготовка будущего приложения), после чего разработчик вставляет в этот шаблон свои операторы на языке C/C++. Затем приложение компилируется в автономный исполняемый файл и может быть запущено на выполнение. Общение с пользователем происходит через специальное так называемое консольное окно, открывающееся средой после запус-

ка приложения (в это окно выводятся сообщения программы, через него вводятся данные для расчета и в него же выводятся результаты расчетов).

Компиляция и сборка проекта осуществляется через команду **Build** главного меню среды. После компиляции и сборки проект можно запустить на выполнение. Запуск на выполнение осуществляется с помощью команды **Debug** главного меню среды.

## Переход к созданию консольного приложения

Для создания консольного приложения необходимо выполнить следующие шаги:

1. Загрузить среду VC++.
2. Выполнить команды главного меню **File|New|Project**. Откроется диалоговое окно, показанное на рис. 1.4. В этом окне выполните последовательно пронумерованные действия. После последнего действия откроется диалоговое окно (рис. 1.5).

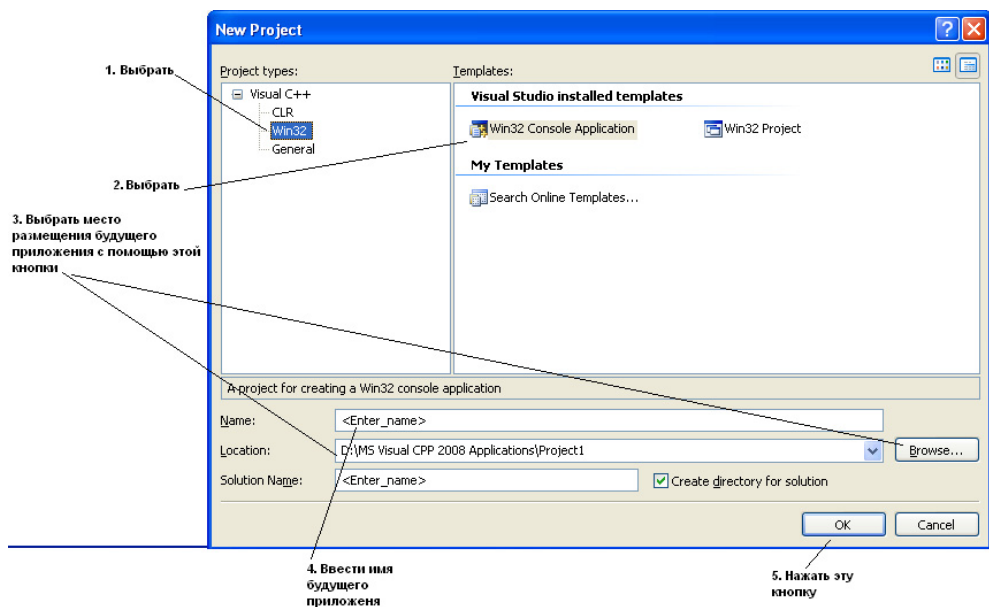


Рис. 1.4. Диалоговое окно **New Project**



Рис. 1.5. Диалоговое окно для определения типа создаваемого проекта

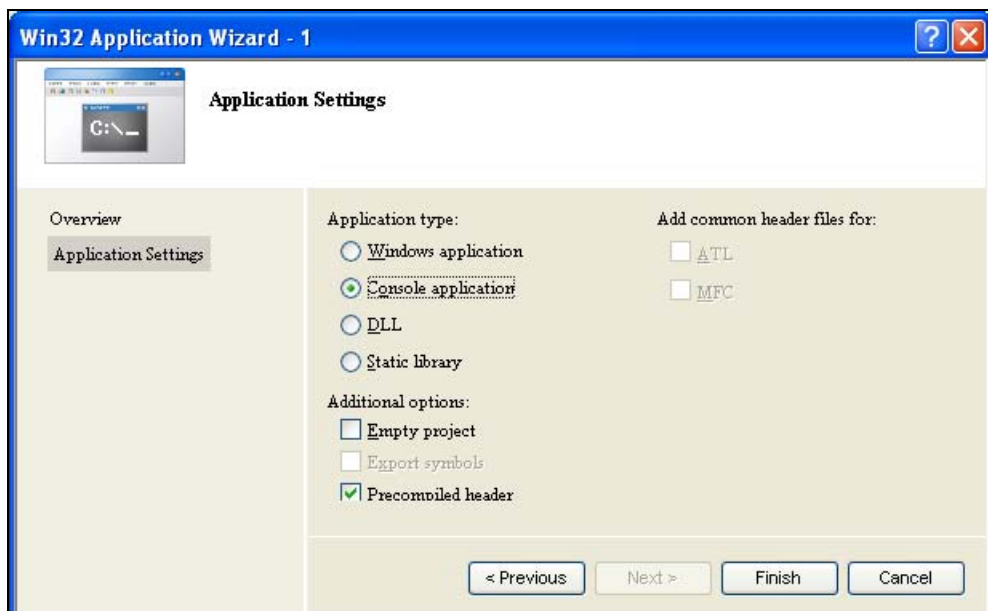


Рис. 1.6. Диалоговое окно для уточнения типа создаваемого проекта

3. В окне, представленном на рис. 1.5, предлагается выбрать тип создаваемого проекта (точнее подтвердить, что вы создаете консольный проект). Для

этого следует нажать на кнопку **Finish**. Дело в том, что среда позволяет создавать множество других типов проектов и задавать к ним дополнительные режимы, поэтому если вы вместо кнопки **Finish** нажмете на кнопку **Next**, то откроется другое диалоговое окно (рис. 1.6).

4. В окне, представленном на рис. 1.6, выбор типа осуществляется с помощью включения радиокнопок (это кнопки в виде окружностей, обладающие тем свойством, что когда одна кнопка включена, то все остальные выключены, что позволяет делать однозначный выбор). Заметьте, что на рисунке включена кнопка для консольного приложения. Если теперь нажать на кнопку **Finish**, то получится такой же результат, как если бы мы нажали на эту кнопку на предыдущем шаге. А результатом будет заготовка консольного приложения (рис. 1.7).

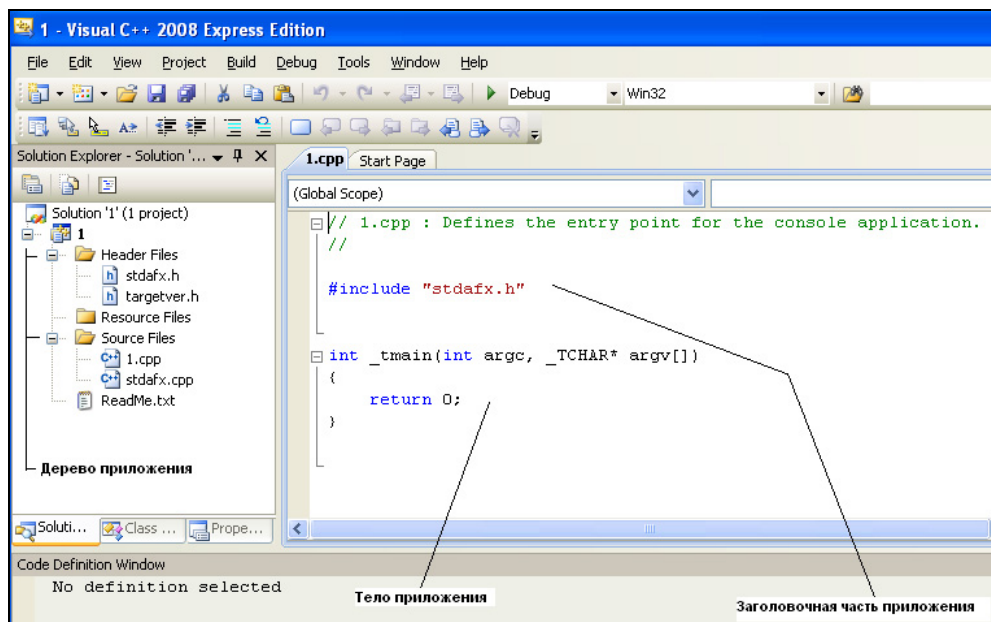


Рис. 1.7. Заготовка консольного приложения

5. Заготовка состоит из заголовка главной функции `int _tmain(int argc, _TCHAR* argv[])` и тела, ограниченного фигурными скобками. Преобразуем заголовок функции `_tmain` к виду `_tmain()`, а из тела удалим оператор `return 0`.

Все это сделаем с помощью Редактора кода, который открывается одновременно с появлением заготовки консольного приложения на экране (заготовка сразу помещается в поле Редактора кода). Чтобы убедиться, что вы находитесь в Редакторе, щелкните кнопкой мыши в любом месте поля заготовки и увидите, что курсор установится в месте вашего щелчка (Редактор ждет в этой точке ваших дальнейших действий). Далее можно набирать любой текст как в обычном современном текстовом редакторе, работать клавишами <Delete>, <Backspace>, клавишами-стрелками и другими необходимыми для ввода и редактирования клавишами.

Мы привели заголовок функции `_tmain` к виду `_tmain()`. Это означает, что наша главная функция не будет иметь аргументов, которые служат для связки консольных приложений. Этим мы заниматься не будем.

Среда VC++ оформляет создаваемое приложение в виде двух контейнеров, вложенных один в другой. Один (главный контейнер) называется Решение (Solution), а другой — Проект (Project). Проект определен как конфигурация (каркас, контейнер), объединяющий группу файлов.

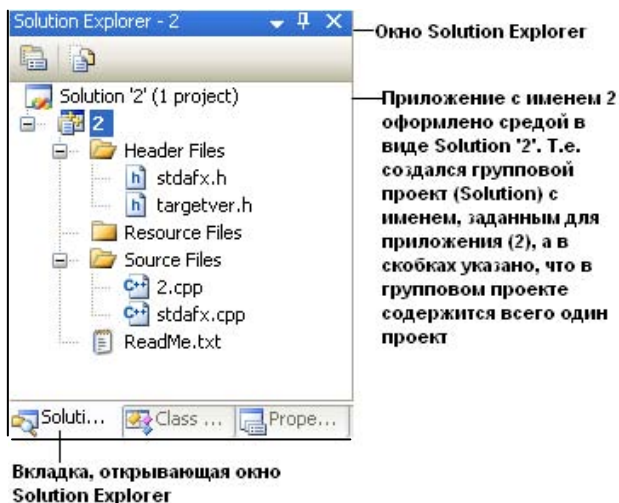


Рис. 1.8. Формирование проекта приложения

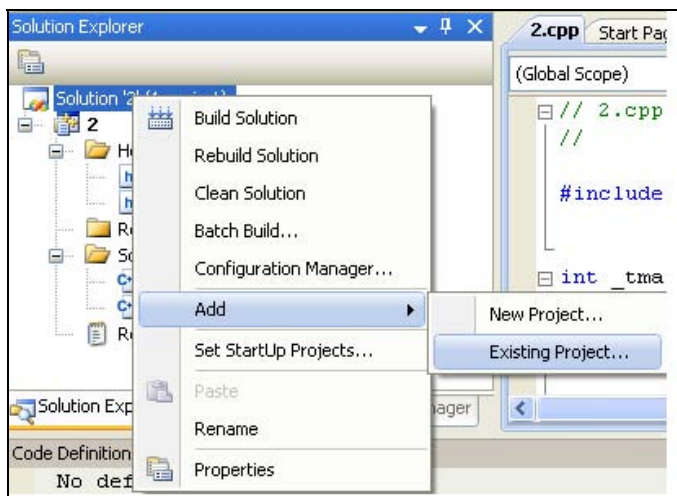
В рамках проекта создается программа, в т. ч. и подлежащая исполнению, т. е. откомпилированная и построенная. Каждый проект содержит по крайней мере две подконфигурации: отладочную и обычную (исполнительскую). Проекты являются частью другого каркаса, другого контейнера, который на-

зывается Solution (решение) и который отражает взаимосвязь между проектами: одно Решение может содержать множество проектов, а проект содержит множество элементов, обеспечивающих существование приложения как такового. Можно сказать, что Решение — это не что иное, как группа объединенных проектов. Назовем его просто: Групповой проект, чтобы термин "Решение" не вводил нас в заблуждение. Существует специальный инструмент работы с групповым проектом, называемый Solution Explorer. К нему можно добраться через опцию **View** меню среды разработки. Сама среда автоматически формирует создаваемое приложение как групповой проект, содержащий собственно проект (это видно из рис. 1.8).

Такой подход к оформлению приложения позволяет работать с группой проектов как с одним целым, что ускоряет процесс разработки приложений. Следует отметить, что не все файлы проекта отображаются в окне **Solution Explorer**.

В качестве примера создадим проект с именем **1** и добавим его к имеющейся группе **2**. Для этого щелкнем правой кнопкой мыши на строке **Solution '2'** и в появившемся контекстном меню выберем команду **Add|Existing Project...** (рис. 1.9).

При этом откроется диалоговое окно для поиска проекта, затем обычным способом откроем проект **1**, в результате чего он добавится к **Solution '2'** (рис. 1.10).



**Рис. 1.9.** Процесс добавления созданного ранее проекта к группе проектов (Solution 2)



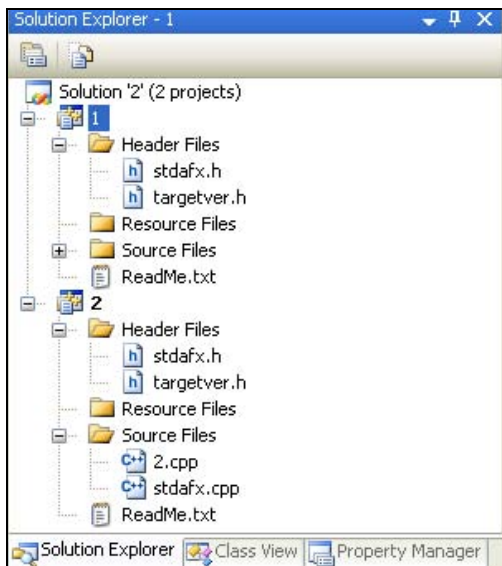


Рис. 1.10. Solution '2' с добавленным к нему проектом 1

Добавленный проект, как и любой другой, входящий в группу проектов, можно удалить. Для этого надо, щелкнув мышью на имени этого проекта, открыть его контекстное меню и выполнить в нем команду **Remove**.

### Примечание

Часто бывает необходимо быстро загрузить ранее сохраненные проекты (не искать их в каталогах). Для этого существует команда главного меню **File|Recent Projects**, которая открывает меню с названиями и путями к ним для ранее выполненных проектов.

Удалим из группы проектов проект **2** и оставим в ней проект **1**. Получим вид рабочего стола среды, показанный на рис. 1.11.

Из рисунка мы видим, что в окне **Recent Projects** (где указаны проекты, с которыми недавно работала среда) имеется список проектов. К каждому из них можно вернуться, если щелкнуть на его имени. Выполнив команду **Remove** (Удалить проект), мы всего лишь удалили сведения о проекте из окна **Solution Explorer**, но не из памяти, поэтому-то и возможно восстановление проекта.

Чтобы удалить проект из памяти, надо найти папку, в которой создан проект, удалить его из папки, а потом щелкнуть на его имени в окне **Recent Projects**,

показанном на рис. 11.1. Если проект из памяти был вами удален, то после щелчка среда выдаст сообщение, что такого проекта нет и попросит подтвердить его удаление. При подтверждении имя проекта удалится из окна **Recent Projects**. Таким способом вы сможете удалять ненужные проекты не только из группы, но и из окна **Recent Projects**, чтобы они вас не раздражали. После всех манипуляций с проектами мы получим созданную заготовку нашего приложения **1**, показанную на рис. 1.12.

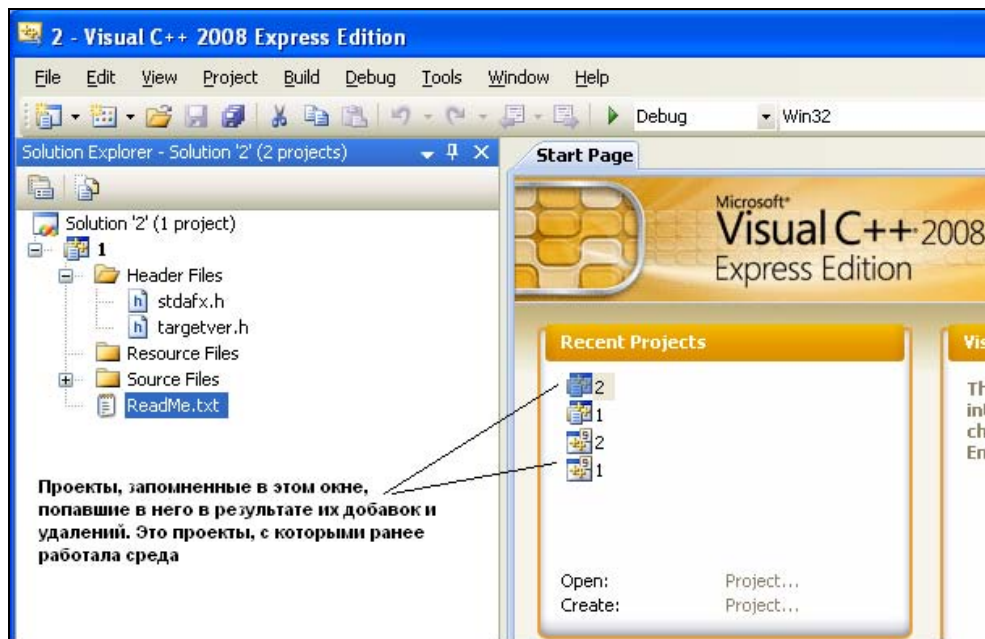


Рис. 1.11. Вид рабочего стола среды после нескольких манипуляций с группой проектов

Мы видим, что над самой заготовкой консольного приложения имеются две вкладки: **1.cpp** и **Start Page**. Если щелкнуть мышью на 1-й из них, то на рабочем столе появится консольная заготовка-проект. Расширение srr означает, что это исходный текст проекта на языке C++. Если щелкнуть мышью на вкладке **Start Page**, то на рабочем столе появится стартовая страница среды (нижняя часть рис. 1.12). Таким способом можно переключаться с одного объекта на другой. Вообще, если вы добавите к проекту еще какой-нибудь элемент, воспользовавшись контекстным меню проекта и выполнив команду **Add** (например, новый srr-файл), то для добавленного элемента тут же на

рабочем столе откроется новая вкладка с именем добавленного элемента, чтобы можно было переключаться между элементами проекта.

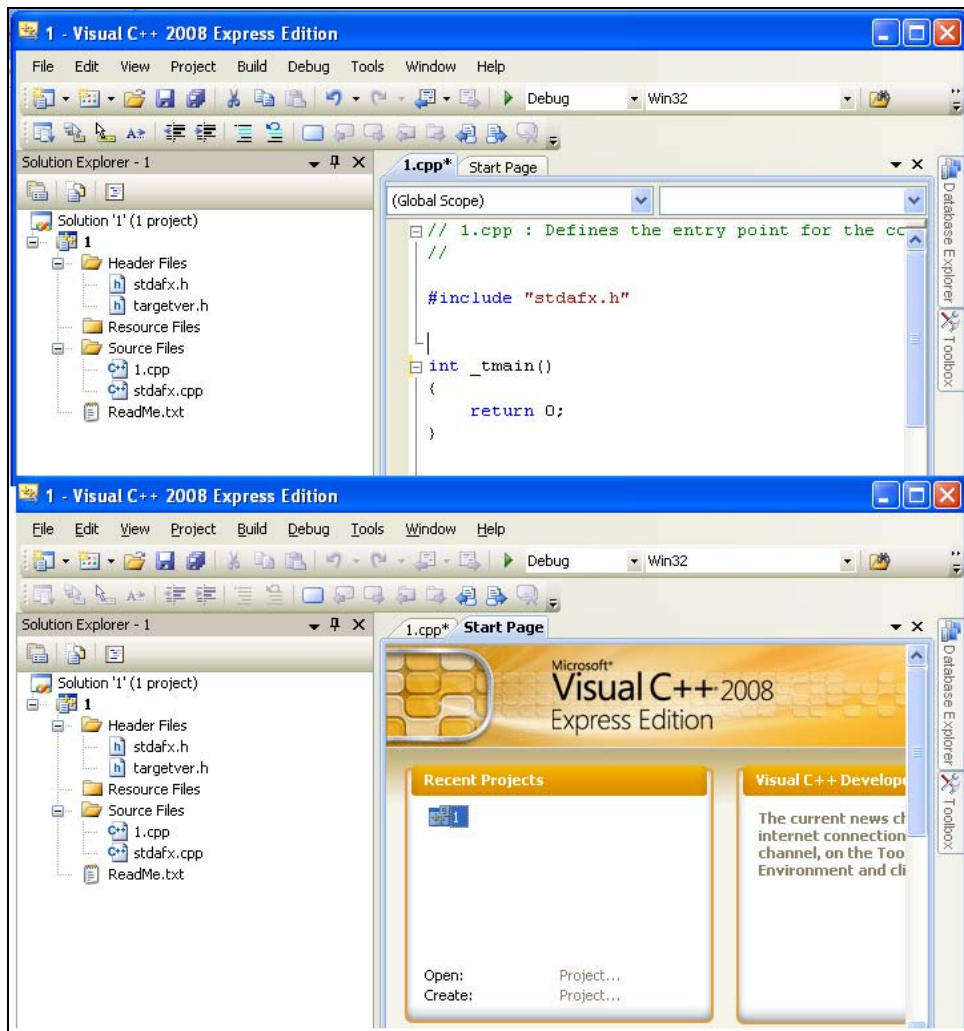


Рис. 1.12. Заготовка консольного приложения и стартовая страница

Теперь посмотрим, что за файлы попали в проект, показанный на рис. 1.12:

- **1.cpp** — это главный исходный файл и точка входа в создаваемое приложение (1 — это в данном случае имя проекта);

- ❑ **stdafx.cpp** — подключает специальный файл Vs9VimTestCpp.pch для компиляции приложения;
- ❑ **stdafx.h** — подключает специальные файлы для компиляции приложения. Вот его содержимое:

```
#pragma once // команды для компиляции:
#include "targetver.h"
#include <stdio.h>
#include <tchar.h>
```
- ❑ **targetver.h** — позволяет использовать специфические свойства Windows Vista. Вот содержимое этого файла:

```
#pragma once //команды для компиляции:
#ifdef _WIN32_WINNT // Allow use of features
//specific to Windows Vista or later.
#define _WIN32_WINNT 0x0600 // Change this to the
//appropriate value to target
//other versions of Windows.
#endif
```
- ❑ **ReadMe.txt** — файл, описывающий некоторые из созданных шаблоном консольного приложения файлов проекта. Посмотреть содержимое файла можно через его контекстное меню, если выполнить в нем команду **Open**.

### Примечание

Мы создали заготовку, построенную по шаблону Win32 Console Application. Но существует и шаблон CLR Console Application, однако мы его не выбрали для построения заготовки. Почему? Этот шаблон предполагает подключение к тексту программы специального пространства System, содержащего классы, которые задают ссылочные типы данных и функции работы с ними. Мы этого аппарата еще не касались и столкнемся с ним в *главе 7*. Там нам и понадобится другой тип консольного приложения, а именно CLR Console Application. Подключение шаблона CLR Console Application к созданию приложения происходит там же, где и подключение к шаблону Win32 Console Application.

## Создание простейшего консольного приложения

Запишем в теле функции `_tmain()` следующие две строки:

```
printf("Hello!\n");
_getch();
```

Это код нашего первого приложения. Он должен вывести на экран текст "Hello!" и задержать изображение, чтобы оно не исчезло, пока мы рассматриваем, что там появилось на экране. Вывод на экран выполняет оператор `printf("Hello!\n");`, а задержку изображения — оператор `_getch();`.

Заметим, что оператором в C/C++ называют некоторое выражение C/C++, оканчивающееся точкой с запятой. В первый оператор входит функция `printf("Hello!\n")`, а во второй — функция `_getch()` (эта функция из C/C++ введена вместо ранее использовавшейся функции `getch()`).

В итоге наше консольное приложение будет иметь вид, представленный на рис. 1.13.

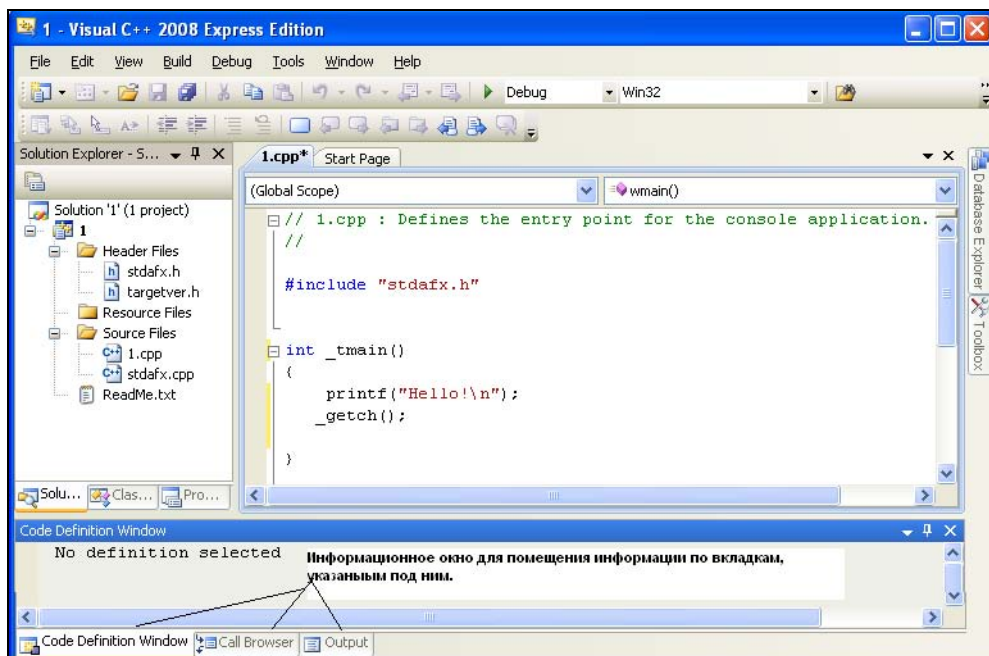


Рис. 1.13. Вид консольного приложения до компиляции

Чтобы приложение заработало, его надо *откомпилировать*, т. е. перевести написанное на языке С в машинные коды. Для этого запускается программа-компилятор. Запускается она либо нажатием клавиши `<F7>`, либо выполнением опции главного меню **Build|Solution**. Если мы проделаем подобные действия, то получим картинку, показанную на рис. 1.14.

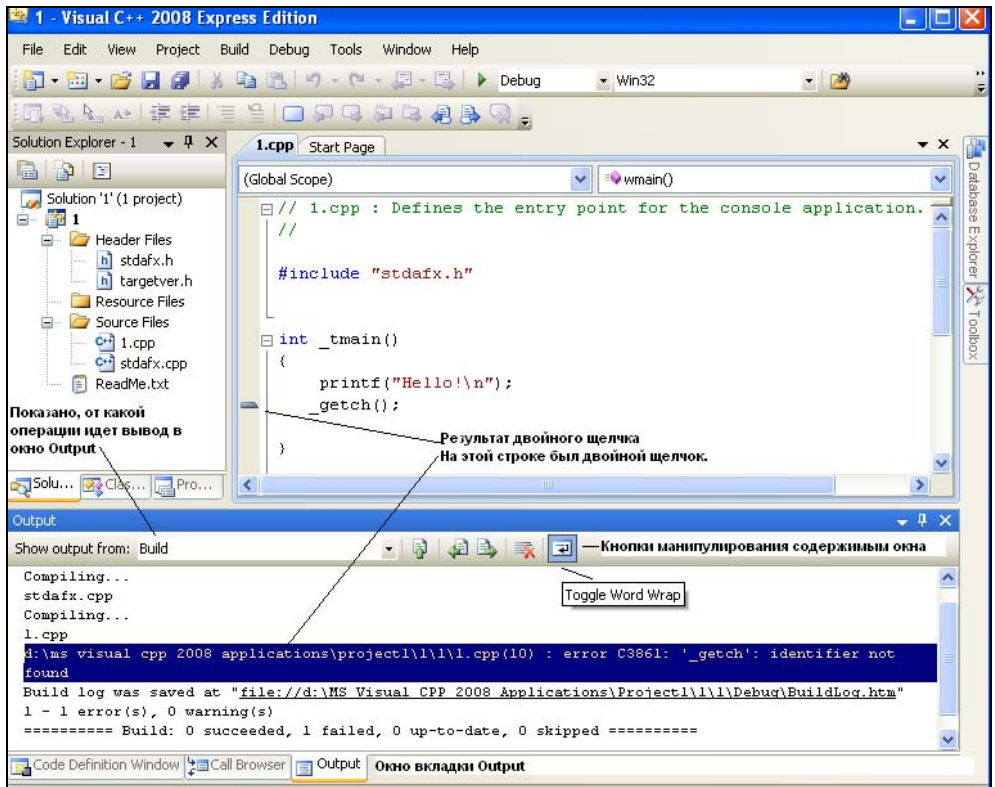


Рис. 1.14. Результат 1-й компиляции приложения

Картинка показывает, что наша компиляция не удалась: в окне вывода высветились сообщения об ошибках. В частности, мы видим сообщение: "error C3861: '\_getch': identifier not found". Это означает, что компилятор не узнает функцию `_getch`.

А почему же компилятор узнал функцию `printf()`? Если посмотреть по справочной системе, то увидим, что местонахождение этой функции находится в файле `stdio.h`. Но если вспомнить, какие файлы автоматически входят в проект при его построении, то увидим, что файл `stdafx.h` как раз и подключает к программе файл `stdio.h`, вот поэтому-то компилятор ведет себя спокойно по отношению к функции `printf()`.

Если кнопкой мыши дважды щелкнуть на каждой строке с информацией об ошибке, то в поле функции `_tmain()`, т. е. в нашей программе в поле подшивки (вертикальная полоса слева от текста) отметится та строка, в которой эта ошибка обнаружена. Этот процесс также показан на рис. 1.14.