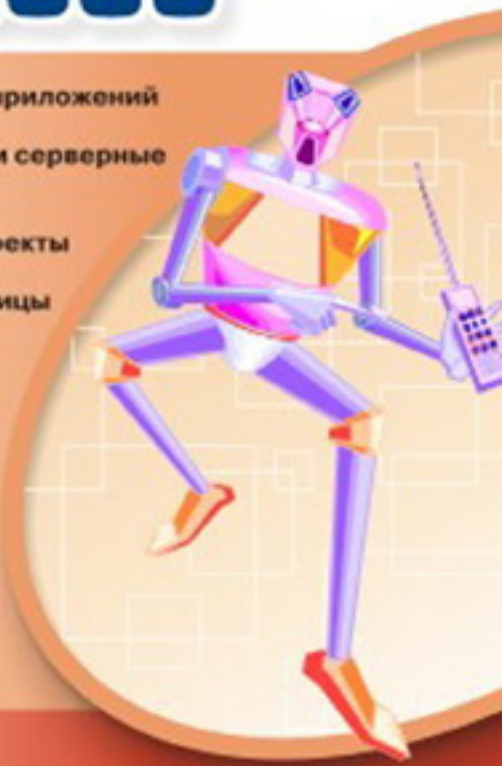
Андрей Гарнаев

Visual Studio .NET 2003

САМОУЧИТЕЛЬ

- Среда разработки приложений
- Пользовательские и серверные компоненты
- Графика и спецэффекты
- Web-, Mobile-страницы и ASP.NET
- Кэширование и аутентификация
- Windows- и Web-службы
- XML, XSL и XPath
- Базы данных и ADO.NET



Более 300 примеров разработки приложений в среде Visual Studio .NET

Андрей Гарнаев

САМОУЧИТЕЛЬ

Visual Studio .NET 2003

Санкт-Петербург

“БХВ-Петербург”

2003

УДК 681.3.068+800.92 Visual Studio
ББК 32.973.26-018.1
Г20

Гарнаев А. Ю.

Г20 Самоучитель Visual Studio .NET 2003. — СПб.: БХВ-Петербург, 2003. — 688 с.: ил.

ISBN 5-94157-336-7

В книге рассматриваются два подхода к разработке приложений на основе Visual Basic .NET и C# средствами технологии .NET — полное создание кода программистом и конструирование приложений с помощью мощных интеллектуальных средств Visual Studio .NET, позволяющих существенно упростить, ускорить и удешевить процесс разработки. Прочитав эту книгу, вы научитесь конструировать приложения для Windows и Web, а также для различных типов переносных устройств, создавать GUI и работать с графикой, разрабатывать пользовательские и серверные компоненты, Windows- и Web-службы, осуществлять работу с базами данных при помощи ADO .NET, работать с объектами ActiveX и COM, а также с XML-документами с использованием XSL и XPath в рамках технологии .NET, конструировать Web-страницы на основе ASP.NET, кэшировать их и аутентифицировать, создавать Mobile-страницы и справочные системы. В книгу включены более 300 примеров, а также справочники по Visual Basic .NET и C#. В основу книги положен курс лекций, читаемый автором в Санкт-Петербургском государственном университете.

Для программистов

681.3.068+800.92 Visual Studio
ББК 32.973.26-018.1

Группа подготовки издания:

Главный редактор	*Екатерина Кондукова*
Зав. редакцией	*Григорий Добин*
Редактор	*Екатерина Капалыгина*
Компьютерная верстка	*Натальи Караваевой*
Корректор	*Елена Самсонович*
Дизайн обложки	*Игоря Цырульникова*
Зав. производством	*Николай Тверских*

Лицензия ИД № 02429 от 24.07.00. Подписано в печать 26.08.03.
Формат 70×100 1/16. Печать офсетная. Усл. печ. л. 55,47.
Тираж 3 000 экз. Заказ №
"БХВ-Петербург", 198005, Санкт-Петербург, Измайловский пр., 29.

Гигиеническое заключение на продукцию, товар № 77.99.02.953.Д.001537.03.02
от 13.03.2002 г. выдано Департаментом ГСЭН Минздрава России.

Отпечатано с готовых диапозитивов
в Академической типографии "Наука" РАН
199034, Санкт-Петербург, 9 линия, 12.

ISBN 5-94157-336-7

Содержание

Предисловие 1

Структура книги 2
Кому адресована эта книга 5
Об авторах 5

Глава 1. Введение 7

Что нового принесла технология .NET 7
.NET Framework 7
Common Language Runtime 8
Языки программирования .NET 8
Ваше первое консольное приложение на VB .NET 9
Параметры утилиты vbc.exe 10
Ваше первое консольное приложение на C# 10
Ваше первое консольное приложение на J# 11
Пространство имен, директива *Imports*, сборки 11
Зачем столько языков 13
Ваше первое консольное приложение, созданное в среде Visual Studio .NET 13
Компиляция сборки 17
Компиляция приложения из командной строки в Visual Studio .NET 17
Компиляция приложения с подключением библиотечного файла 18
Классы 19
Пространства имен, загружаемые в Visual Studio .NET по умолчанию 22
Статические поля, свойства и методы 22
Создание библиотек классов 24
Наследование 25

События 28
Структурная группировка кода 31
Интерфейсы 32
Применение стандартных интерфейсов 35
Межъязыковое наследование 38
Генерация кода и исполняемого файла самой программой 41
Вызов программы из кода 43
Сериализация и десериализация объектов 43
Поток 47
Приостановка и уничтожение потока 48
Приоритет выполнения потоков 49

Глава 2. Интегрированная среда разработки 51

Стартовая страница Visual Studio .NET 51
Создание проекта 52
Файлы проекта 53
Открытие проекта 54
Строка меню 54
Окно *Solution Explorer* 55
Добавление ссылок в проект 56
Окно конструирования формы 56
Изменение размеров и местоположения формы и элементов управления 57
Удаление элементов управления 57
Копирование и вставка элементов управления 57
Окно *Toolbox* 58
Сетка в форме 58
Форматирование группы элементов управления 58
Форматирование группы элементов управления командой *Format* 59
Окно *Properties* 60
Окно редактирования кода 61
Управление структурой редактора кода 62
Раскрывающиеся списки редактора кода 63
Поиск в коде 63
Обозначение кода закладками 64
Интеллектуальные возможности редактора кода 64
Использование справочной системы 67
Управление интеллектуальными возможностями редактора кода 67

Установка и снятие знаков комментариев у выделенного блока кода 68
Окно *Task List* 68
Окно *Object Browser* 68
Окно *Class View* 69
Окно *Output* 70
Настройка среды разработки 70
Макросы 71
Добавление макроса в пользовательскую панель инструментов 72
Редактирование кода макроса 72

Глава 3. Windows-приложения 75

Создание формы в Visual Studio .NET 75
Разбор кода первого Windows-приложения 78
Создание формы в коде 79
Создание пользовательского значка формы 81
Добавление ссылки на значок в коде 82
Программное закрытие формы 82
Как убрать значок выполняемого приложения из строки состояния Windows 83
Прозрачная форма 83
Создание "пасхального яйца" 84
Задание местоположения формы при ее запуске 86
Задание размеров формы 86
Задание фонового рисунка свойством *BackgroundImage* 86
Задание фонового рисунка его прорисовкой 88
Форма произвольной конфигурации 89
Подвижная форма переменной конфигурации 91
Обмен сообщениями между двумя формами 94
Еще раз об обмене сообщениями между формами 96
Наследование форм 97
Построение меню *(MainMenu)* 100
Построение меню в коде 104
Контекстное меню *(ContextMenu)* 106
Построение контекстного меню в коде 107
Панель инструментов *(ToolBar)* 109
Панель инструментов с кнопками и разделителем 109
Построение в коде панели инструментов с кнопками и разделителем 111
MDI-форма 112

Глава 4. Элементы управления 119

Окно *Toolbox* 119
Надпись *(Label)* 122
Прозрачная надпись 123
Надпись с гиперссылкой *(LinkLabel)* 123
Кнопка *(Button)* 125
Свойства, устанавливающие местоположение и размер элементов управления в форме в коде 126
Задание местоположения и размеров элементов управления в коде 126
Курсор 127
Создание пользовательского курсора в Visual Studio .NET 128
Вывод изображения на поверхность элемента управления 130
Поле *(TextBox)* 131
Флажок *(CheckBox)* 133
Переключатель *(RadioButton)* 134
Рамка *(GroupBox)* 135
Всплывающая подсказка *(ToolTip)* 137
Графическое поле *(PictureBox)* 138
Задание ссылки на файл в коде и поворот рисунка 139
Панель *(Panel)* 140
Таймер *(Timer)* 140
Динамические свойства 141
Список *(ListBox)* 144
Заполнение списка в коде 146
Создание пользовательского списка 146
Поле со списком *(ComboBox)* 149
Список с флажками *(CheckedListBox)* 151
Горизонтальная *(HScrollBar)* и вертикальная *(VScrollBar)* полосы прокрутки 152
Счетчик с числовым полем ввода *(NumericUpDown)* 154
Счетчик с текстовым полем ввода *(DomainUpDown)* 156
Ползунок *(TrackBar)* 157
Индикатор прогресса *(ProgressBar)* 159
Указатель времени и даты *(DateTimePicker)* 161
Календарь *(MonthCalendar)* 162
Список изображений *(ImageList)* 164
Создание списка изображений в коде 166

Диалоговое окно *Open (OpenFileDialog)* ... 167
Чтение текстового файла ... 169
Проблема русификации ... 169
Загрузка файлов из Интернета ... 170
Определение IP-адреса компьютера ... 171
Диалоговое окно *Save As (SaveFileDialog)* ... 171
Диалоговое окно *Browse For Folder (FolderBrowserDialog)* ... 173
Диалоговые окна *Font (FontDialog)* и *Color (ColorDialog)* ... 176
Строка состояния *(StatusBar)* ... 179
Разделитель *(Splitter)* ... 182
Извещающий значок *(NotifyIcon)* ... 185
Обработчик ошибок *(ErrorProvider)* ... 186
Вкладки *(TabControl)* ... 190
Элемент управления *ListView* ... 193
Операция *Drag and Drop* для элементов управления *ListView* ... 197
Элемент управления *TreeView* ... 200
Простейший браузер на основе *TreeView* ... 200
Элемент управления *FileSystemWatcher* и отслеживание изменений файловой системы ... 203

Глава 5. Графика ... 207

Вывод текста ... 207
Двойная буферизация ... 209
Вращающаяся и перемещающаяся фигура ... 209
Повернутая надпись ... 213
Круги на воде ... 214
Еще раз об избежании мерцания при перерисовке формы ... 217
Цветовые фильтры ... 219
Наложение изображений ... 224
Создание изображения в памяти и вывод его в файл ... 226
Операция Drag and Drop для графического файла ... 227
Как в графическое поле вставить содержимое буфера обмена ... 229

Глава 6. Создание библиотек классов и компонентов ... 231

Расширение существующего элемента управления ... 231
Создание новых свойств у элемента управления ... 231
Переопределение событий ... 234

Создание пользовательских событий ... 235
Создание DLL-файла ... 236
Создание новой вкладки панели инструментов *Toolbox* и добавление на нее значка созданного элемента управления ... 237
Настройка панели инструментов *Toolbox* ... 238
Тестирование DLL-файла ... 239
Атрибуты ... 239
Кнопка переменной формы ... 240
Редактируемый элемент управления *ListView* ... 245
Создание пользовательского элемента управления на основе класса *UserControl* ... 249
Создание пользовательского элемента управления на основе класса *Control* ... 253
Перерисовка существующего элемента управления ... 257

Глава 7. ADO .NET ... 263

Соединение с источником данных ... 263
Класс *DataSet* ... 264
Адаптер ... 264
Таблица ... 264
Связанные элементы управления ... 264
Связывание данных с элементом управления *DataGrid* в Visual Studio .NET ... 265
Установка соединения с базой данных на этапе конструирования ... 265
Создание адаптера на этапе конструирования ... 266
Создание объекта *DataSet* на этапе конструирования ... 270
Заполнение элемента управления *DataGrid* на этапе конструирования ... 271
Заполнение элемента управления *DataGrid* в коде ... 272
Выборка записей с помощью объекта *DataTable* ... 272
Создание объекта *DataTable* в коде ... 273
Связывание данных со списком в Visual Studio .NET ... 274
Связывание данных со списком в коде ... 275
Простое связывание ... 276
Браузер базы на основе простого связывания ... 277
Согласованная работа списка и элемента управления *DataGrid* ... 281
Отбор данных по критерию ... 283

Запрос и язык запросов SQL ... 285
Оператор *SELECT* ... 286
Возвращение всех записей таблицы ... 286
Возвращение всех записей одного поля ... 286
Возвращение всех записей двух полей таблицы ... 286
Упорядочивание элементов и предложение *ORDER BY* ... 286
Упорядочивание записей по одному полю ... 286
Упорядочивание записей по двум полям ... 286
Выборка записей по критерию и предложение *WHERE* ... 287
Ссылка на строки и даты в операциях сравнения ... 287
Предложение *WHERE* и выборка записей по простому критерию ... 287
Выборка записей по составному критерию ... 287
Выборка записей, значения специфицированного поля которых принадлежат указанному диапазону или лежат за его пределами ... 287
Выборка записей, значения специфицированного поля которых принадлежат указанному множеству ... 288
Выборка записей по шаблону ... 288
Выбор различных записей ... 288
Нахождение общего числа записей, возвращаемого запросом ... 288
Нахождение общего числа не пустых записей ... 289
Статистические функции ... 289
Определение, пусто ли поле, и значение *NULL* ... 289
Оператор *DELETE* ... 290
Оператор *INSERT* ... 290
Оператор *UPDATE* ... 290
Создание групп записей ... 290
Создание псевдонима с использованием предложения *AS* ... 291
Предложение *HAVING* ... 291
Обращение к данным с помощью ADO .NET ... 291
Удаление записи командой *DELETE* ... 292
Вставка записи командой *INSERT* ... 293
Модификация данных командой *UPDATE* ... 293
Поиск общего числа записей командой *SELECT* ... 294
Выборка записей командой *SELECT* ... 295
Использование параметров SQL-команды ... 296
Встроенные процедуры ... 298
Транзакция ... 301
Заполнение элемента управления *ListView* из базы данных ... 303

Сортировка и фильтрация с помощью объекта *DataTable* 305
Запись данных SQL-запроса в XML-документ 306

Глава 8. Отчет Crystal Report .NET и печать данных 309

Создание отчета в Crystal Report .NET 309
Панели инструментов Crystal Report .NET 322
Печать данных 323
Элемент управления *PrintDialog* 323
Элемент управления *PrintPreviewDialog* 323
Элемент управления *PageSetupDialog* 323
Элемент управления *PrintDocument* 324
Предварительный просмотр и печать текстового файла 325

Глава 9. XML, XSL и Xpath 331

Синтаксис XML 331
Как выглядит XML-документ в браузере 333
Атрибуты 334
Применение каскадной таблицы стилей для форматного вывода текста 335
XSL 337
Элемент *<xsl:for-each>* 340
Выборка элементов по ключу 341
Элемент *<xsl:if>* 342
Элемент *<xsl:choose>* 343
Элемент *<xsl:sort>* 344
Элемент *<xsl:number>* 346
XPath 347
Чтение атрибутов 349
Работа со строками 351
XSD 352
XML и .NET Framework 353
Создание графического файла на основе данных, хранящихся в XML-документе 354
Считывание XML-данных из строки 357
Абстрактный класс *XmlReader* 358
Считывание данных из XML-документа с помощью класса *XmlNodeReader* 358

Работа с атрибутами ... 360
Считывание и запись данных в XML-документ
с использованием объекта *DataSet* ... 365
Поиск записи в XML-документе ... 367
Представление набора данных в виде кода XML ... 368
Запись данных в XML-документ с использованием
класса *XmlTextWriter* ... 368
Запись данных из базы данных в XML-документ ... 371
Выборка данных с помощью команд языка XPath ... 371
Выборка данных по атрибуту ... 373

Глава 10. Компоненты ASP .NET ... 375

Создание первого приложения ASP .NET в Visual Studio .NET ... 375
Создание виртуального каталога на сервере IIS ... 378
Создание первого приложения ASP .NET в коде ... 378
Импортирование пространства имен ... 380
Директивы страницы ... 380
Передача данных из HTML-формы ... 381
Передача данных из строки запроса ... 382
Жизненный цикл Web-страницы ... 383
Как определить, была ли загружена страница ... 384
Передача построенных на сервере изображений ... 386
Web-формы и серверные элементы управления ... 388
Серверные элементы управления *HTML* ... 389
Загрузка данных от клиента на сервер ... 390
Серверные элементы управления *Web* ... 392
Надпись ... 393
Кнопка ... 394
Событие *Command* ... 396
Отделение кода от представления ... 397
Поле ... 399
Флажок ... 401
Список флажков ... 402
Свойство *AutoPostBack* и немедленная отправка данных на сервер ... 403
Переключатель ... 403
Список переключателей ... 405
Заполнение списка переключателей из XML-документа ... 406

Список ... 407
Выбор нескольких элементов из списка ... 408
Раскрывающийся список ... 410
Заполнение списка из динамического массива ... 411
Заполнение списка из базы данных ... 412
Установка стиля элемента управления ... 413
Style Builder ... 415

Глава 11. Пользовательские Web-компоненты ... 477

Пользовательский элемент управления ... 477
Создание пользовательского элемента ... 477
Регистрация пользовательского элемента управления ... 478
Поля пользовательского элемента управления ... 479
Свойства пользовательского элемента управления ... 480
Методы пользовательского элемента управления ... 482
События пользовательского элемента управления ... 483
Создание Web-компонента, расширяющего существующий компонент ... 485
Создание пользовательского элемента управления в Visual Studio .NET ... 487
Изменение префикса, используемого по умолчанию в Visual Studio .NET ... 491
Изменение значка элемента управления, используемого по умолчанию в Visual Studio .NET ... 491
Создание элемента управления, имеющего дочерние элементы ... 492

Глава 12. Приложения ASP .NET ... 497

Отсылка сообщений по электронной почте ... 497
Отсылка сообщений по электронной почте в HTML-формате ... 498
Отсылка сообщений по электронной почте с приложенным файлом ... 498
Проверка пароля и имени пользователя ... 499
Второй способ проверки пароля и имени пользователя ... 501
Сессии ... 501
Запрет открытия страницы без посещения предыдущей ... 504
Регистрация приложения в IIS ... 505
Продолжительность сессии ... 505
Выполнение сессии в связанном режиме ... 506
Выполнение сессии во внепроцессорном режиме ... 507

События приложения и сессии ... 507
Учет числа посетителей страницы с установкой продолжительности сессии ... 508
Простая аутентификация клиента ... 509
Cookies ... 512
Удаление cookies в указанное время ... 514
Конфигурирование приложения ... 516
Чтение параметров конфигурации ... 518
Кэширование ... 519
Кэширование на основе строки запроса ... 520
Кэширование фрагмента страницы ... 522
Кэширование фрагмента страницы с учетом свойств ... 523
Кэширование объекта ... 525
Локализация приложения ... 526
Применение файлов ресурсов при локализации приложения ... 528

Глава 13. Web- и Windows-службы ... 531

Web-службы ... 531
Ваша первая Web-служба средствами SDK .NET ... 532
Тестирование Web-службы, созданной средствами SDK .NET ... 533
Передача объектов .NET ... 536
Протоколы доступа к Web-службам ... 537
Создание клиента Web-службы ... 538
Синхронный и асинхронный вызовы Web-службы ... 543
Ваша первая Web-служба, построенная средствами Visual Studio .NET ... 544
Создание клиента Web-службы средствами Visual Studio .NET ... 547
Web-служба доступа к данным ... 549
Безопасность Web-служб ... 553
Windows-службы ... 556
Ваша первая Windows-служба ... 557

Глава 14. Программирование мобильных устройств ... 561

Первое приложение Pocket PC ... 561
WML ... 565
Первая мобильная Web-форма ... 568
Создание мобильного интерфейса в браузере ... 570

Создание мобильной Web-формы в Visual Studio .NET 571
Постраничный вывод данных 574
Несколько форм на мобильной странице 575
Вывод изображений 577
Телефонный звонок 578

Глава 15. Обработка ошибок и отладка программ 579

Разработка процедур, предотвращающих появление ошибок 579
Оператор *Try — Catch — Finally* 582
Обработка специальных исключений 583
Контроль вводимых значений с помощью обработки события *KeyPress* 584
Проверка вводимых данных с помощью регулярных выражений 585
Отладка программ 585
Отображение нумерации строк 586
Ошибки компиляции 586
Ошибки выполнения 587
Логические ошибки 589
Точка прерывания 589
Пошаговое выполнение программ 591
Исследование переменных 591
Окно *Output* 593
Обработка ошибок в ASP .NET 594

Глава 16. Windows API и COM 597

Windows API 597
Проигрывание WAV-файла 598
Форма, двигающаяся внутри активного окна 599
COM 600
Нахождение значений арифметических выражений 600
Отсылка сообщений по почте, используя MS Outlook 602
Проигрывание звуковых файлов
с помощью Microsoft Multimedia Control 603
Проигрыватель CD 605

Глава 17. Справочная система 607

Этапы построения справочной системы 607
Создание разделов справочной системы 607

Построение таблицы содержания ... 608
Конструирование файла проекта ... 610
Создание индекса ... 615
Элемент управления *HelpProvider* ... 615
Справка по элементу управления ... 616
Отображение справки при нажатии кнопки ... 616
Отображение кнопки *Help* ... 616
Кнопка *Help*, клавиша <*F1*> и всплывающая подсказка по элементам управления ... 617

Глава 18. Настройка конфигураций и развертывание приложений ... 619

Мастер Setup Wizard ... 620
Установка приложения ... 624
Удаление приложения ... 626
Создание ярлыка приложения на рабочем столе ... 627

Глава 19. Справочное руководство по Visual Basic .NET ... 631

Типы данных ... 631
Объявление переменных ... 631
Комментарии ... 632
Массивы ... 632
Динамические массивы ... 633
Константы ... 633
Явное и неявное преобразование типов ... 633
Выражения и операторы ... 634
Операторы присвоения ... 635
Операторы управления ... 636
Оператор условного перехода ... 636
Оператор выбора ... 637
Оператор цикла *For — Next* ... 637
Оператор цикла *For Each* ... 638
Оператор цикла *While — Wend* ... 639
Оператор цикла *Do — Loop* ... 639
Оператор безусловного перехода *GoTo* ... 640
Передача параметров по ссылке и значению ... 641
Случайные числа ... 641

Глава 20. Справочное руководство по C# 643

Типы данных и объявление переменных 643
Явное и неявное преобразование типов 644
Комментарии 645
Массивы 645
Константы 646
Выражения и операторы 646
Операторы присвоения 646
Блоки 647
Операторы управления 647
Оператор *if* 647
Оператор *switch* 648
Оператор *for* 649
Оператор *foreach* 650
Оператор *while* 651
Оператор *do* 651
Оператор *return* 651
Операторы *break* и *continue* 653
Оператор *goto* 653
Операторы *try* и *throw* 654
Перечисления 655
Структуры 656
Классы 657
Методы и свойства 658
Наследование 659
Интерфейсы 663
События и делегаты 664

Предисловие

Технология программирования .NET, а также среда разработки приложений на базе этой технологии Visual Studio .NET 2003 разделила эпоху программирования на два этапа — до и после .NET. Конечно, до появления данной технологии в мире происходило множество важных событий: создание операционной системы Windows, визуальный подход к разработке приложений и т. д. Все они были предтечей самого главного события — появления технологии программирования .NET, а также Visual Studio .NET 2003 — среды конструирования приложений на базе этой технологии. В корпорации Microsoft обратили внимание на то, что разработка современных программных продуктов требует огромных временных, а следовательно, ресурсных затрат. Кроме того, имелось огромное количество типов приложений (например, приложений для Windows, Web, переносных устройств), создание каждого из которых требовало специальных средств их конструирования, подготовки программистов соответствующей специализации, также требующей огромных временных и денежных затрат. Корпорация Microsoft поставила перед собой революционную задачу — создание универсальной технологии программирования, которую с успехом можно применять при разработке любого типа приложений, технологии, независимой от платформы, на которой будет разворачиваться приложение. Visual Studio .NET 2003 — это универсальная среда разработки всех типов приложений на базе .NET на основе единого визуального интерфейса. Сегодня нет нужды изучать десятки различных языков программирования и различные среды разработки. Любое приложение можно создать в рамках .NET с помощью единого инструментария — Visual Studio .NET 2003.

Читатель, который только начинает знакомиться с технологий .NET, сразу же обращает внимание на большое число .NET-языков программирования: Visual Basic .NET, C#, J# .NET, JScript .NET. Это еще не все .NET-языки, грядет появление новых. Кроме того, благодаря разработанной в корпорации Microsoft технологии .NET, сейчас сам процесс построения трансляторов новых языков программирования весьма упрощен и каждый пользователь может создать свой собственный .NET-язык. Зачем же нужно так много

языков, если они все опираются на объекты .NET Framework, а следовательно, сконструированные на их основе приложения будут равносильны. В основном это связано с тем, что до появления технологии .NET уже было много языков. Поэтому тот, кто программировал раньше на Visual Basic, VBA или VBScript, скорее всего выберет Visual Basic .NET. Тот, кто работал с C++ или Java, предпочтет C# или J# .NET, а с JavaScript или JScript — JScript .NET.

Создавать приложения .NET можно двумя способами: либо вручную, набирая весь код, а затем компилируя его с командной строки, либо с применением среды разработки Visual Studio .NET 2003. При первом способе можно создать очень компактный код, но это требует от программиста больших временных затрат. Второй способ, благодаря наличию мощных интеллектуальных средств Visual Studio .NET 2003, а также большому количеству встроенных мастеров кода, существенно ускоряет весь процесс разработки проекта: от его создания и отладки до создания дистрибутива. Недостатком второго способа является то, что код, сконструированный мастерами проекта, бывает немного громоздким, а в сам проект добавляются файлы, которые нужны не проекту, а среде разработки.

В данной книге рассматриваются оба способа — первый для того, чтобы читатель мог изучить саму технологию .NET, а второй — чтобы научиться быстро создавать приложения .NET. Прочитав эту книгу, читатель сможет на основе Visual Basic .NET и C# конструировать Windows-, Web-приложения, а также приложения для переносных устройств типа мобильного телефона, Smart Phone, Pocket PC. Вы научитесь конструировать GUI, работать с графикой, создавать пользовательские и серверные компоненты, конструировать Windows- и Web-службы, работать с базами данных с помощью ADO .NET, с XML-документами — с использованием XSL и XPath в рамках технологии .NET, конструировать Web- и Mobile-страницы, кэшировать их и аутентифицировать, а также работать с объектами ActiveX и COM.

Структура книги

По своей структуре книга состоит из 20 глав. В *первой* главе описано, что такое технология .NET, что она принесла нового, описано, как создаются консольные приложения на Visual Basic .NET, C#, J# .NET. Изложен объектно-ориентированный подход, применяемый в приложениях .NET: создание классов, интерфейсов, делегатов, реализация событий и наследования, осуществление межъязыкового наследования, работа потоков, сериализация и десериализация объектов, возможность создания программ другими программами.

Во *второй* главе приведена структура интегрированной среды разработки приложений Visual Studio .NET 2003.

Третья глава посвящена технологии построения Windows-приложений: конструированию формы в Visual Studio .NET и в коде, стационарной и подвижной формы переменной конфигурации, дочерней формы, обмену сообщениями между формами, созданию главного и контекстного меню, строки состояния, панели инструментов и MDI-формы.

Четвертая глава содержит описание коллекции элементов управления пространства имен `System.Windows.Forms`. Показано, как задаются динамические свойства, конструируется пользовательский список, производится работа с текстовыми файлами и решается проблема русификации, определяется IP-адрес компьютера, программируются операции Drag and Drop, отслеживаются изменения файловой системы.

В *пятой* главе изложены основные сведения о работе с текстом, реализации двойной буферизации, осуществлении перемещения и вращения изображений, создании цветовых фильтров, проведении операции Drag and Drop для графического файла, вставке в графическое поле содержимого буфера обмена.

В *шестой* главе дается представление о расширении существующего элемента управления, конструировании новых свойств и методов, переопределении старых и создании новых событий, конструировании и тестировании DLL-файла, создании элемента управления на основе классов `UserControl` и `Control`. Показано как перерисовывать существующий элемент управления.

В *седьмой* главе рассматривается технология работы с базами данных средствами ADO .NET. Показано, как производить соединение с источником данных, как строится адаптер, осуществляется связывание элементов управления, а также конструирование браузера данных, отбор данных по критерию, применение параметров в SQL-командах, создание встроенных процедур и реализация транзакций.

Восьмая глава описывает работу с пакетом Crystal Report .NET как инструментальным средством конструирования отчетов. Он осуществляет выборку и форматирование результирующего набора данных из базы данных или другого источника данных, а также предоставляет возможность создавать любые отчеты: от простого табличного до сложного, использующего многочисленные диаграммы, таблицы и производящего сложные вычисления над ними.

В *девятой* главе речь пойдет о XML, XSL, XPath и их применении в .NET. Приведены краткие справочные сведения по работе с XML, XSL и XPath, проиллюстрированные примерами. Показано, как конструируется графический файл на основе данных, хранящихся в XML-документе. Представлена реализация считывания данных из XML-документа с использованием класса `XmlNodeReader`, считывание и запись данных в XML-документе с помощью объекта `DataSet`, поиск записи в XML-документе, запись данных в XML-документ с применением класса `XmlTextWriter`, выборка данных с помощью команд языка XPath и по атрибуту.

В *десятой* главе рассматривается создание приложений ASP .NET и их компоненты.

ASP .NET предоставляет в распоряжение разработчика не только большую коллекцию элементов управления, но позволяет также создавать как пользовательские, так и Web-компоненты. В *одиннадцатой* главе описана технология создания таких компонентов, рассказано, как эти компоненты регистрируются, как у них определяются поля, свойства, методы и события. Как с их помощью расширяются уже существующие компоненты, конструируются элементы управления, содержащие дочерние элементы.

В *двенадцатой* главе рассмотрены некоторые нюансы построения приложений ASP .NET: конфигурирование приложения, регистрация приложения в IIS, аутентификация клиента, кэширование, локализация приложения, файлы ресурсов, отсылка сообщений по электронной почте, сеансы, cookies, запрет открытия страницы без посещения предыдущей.

Тринадцатая глава содержит сведения о Web- и Windows-службах.

Visual Studio .NET позволяет создавать приложения для Pocket PC и различных переносных устройств типа Smart Phone, мобильный телефон по той же схеме, как и для обычных Windows- и Web-приложений. В *четырнадцатой* главе приведены примеры подобных приложений, рассмотрено конструирование мобильных форм, создание мобильного интерфейса в браузере, осуществление постраничного вывода данных, вывод изображений, телефонный звонок.

В главе *пятнадцатой* приводится описание способов обработки ошибок и отладки программ.

Хотя по своей идеологии .NET разрабатывалась как платформенно независимая технология, тем не менее для нее остаются достижимыми и те богатства средств разработки приложений, которые были собраны в Windows API, а также созданы на основе технологии COM. В *шестнадцатой* главе рассказано, как производится вызов Windows API-функций, с помощью которых можно проигрывать WAV-файлы и сконструировать форму, двигающуюся внутри активного окна, как можно интегрировать возможности офисных приложений с помощью технологии COM в .NET-проекты, а также как можно сконструировать простейший проигрыватель CD.

В *семнадцатой* главе описано то, как создается справочная система.

В главе *восемнадцатой* рассматривается настройка конфигурации и развертывание приложения.

В главах *девятнадцатой* и *двадцатой* предложены справочные руководства по Visual Basic .NET и C#.

Кому адресована эта книга

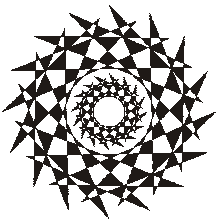
Книга не предполагает какого-либо опыта разработки проектов в рамках технологии .NET у читателя. Благодаря наличию более 300 примеров, справочным разделам по Visual Basic .NET и C#, доскональному обзору материала, данная книга может быть полезной широкому кругу читателей, изучающих эти языки программирования, а также тем из них, которые хотят научиться конструировать современные приложения различных типов, создавая их вручную или с помощью средств Visual Studio .NET 2003, существенно ускоряющей и упрощающей весь процесс работы программиста.

Об авторах

Андрей Юрьевич Гарнаев (**garnaev@ag2784.spb.edu**) — доктор физико-математических наук, профессор Санкт-Петербургского государственного университета. Область интересов: технологии программирования, информационные системы и математическая кибернетика. Им опубликовано двенадцать монографий. Является автором книг по программированию на Visual Basic — "*Visual Basic 6.0: разработка приложений*", *"Visual Basic, NET: разработка приложений*", по офисному программированию — *"Excel 2002: разработка приложений*" и *"Excel*, *VBA и Internet в экономике и финансах*" издательства "BHV-Петербург".

Автор благодарит вас за выбор этой книги и надеется, что она сможет стать вашим надежным гидом в захватывающем и сказочно удивительном новом мире — программировании в .NET вручную или с помощью средств Visual Studio .NET 2003.

Глава 1



Введение

В данной главе описана технология .NET. Рассматривается создание консольных приложений на Visual Basic .NET, C#, J# .NET. Изложен объектно-ориентированный подход, применяемый в приложениях .NET: создание классов, интерфейсов, делегатов, реализация событий и наследование, осуществление межъязыкового наследования, работа потоков, сериализация и десериализация объектов. Представлено описание того, как программы могут порождать другие программы.

Что нового принесла технология .NET

Мы все стали свидетелями появления новой революционной технологии программирования .NET, предложенной корпорацией Microsoft. Что же нового принесла эта технология?

- Пришествие .NET приведет к крупным изменениям на уровне парадигм Windows-программирования.
- .NET ставит программистов VB в равные условия с программистами C++ в отношении широты возможностей языка.
- .NET превращает программистов VB в разработчиков Internet-приложений, не требуя от них практически никаких усилий.
- .NET предоставляет в распоряжение программистов огромную библиотеку объектов, ускоряющую процесс разработки и расширяющую его возможности.

Архитектура .NET исправляет два основных недостатка COM: проблемы контроля версии и размещения, а также утечки памяти в тех случаях, когда программист забывает своевременно освобождать объекты в своей программе.

.NET Framework

.NET Framework — это набор объектов и планов (blueprints, т. е. описание объектов), созданных корпорацией Microsoft для разработки приложений

(Windows и Internet). Такие объекты размещаются внутри логических групп, которые называются пространствами имен.

Common Language Runtime

Common Language Runtime (CLR, общая для языков среда исполнения) — это среда, которая управляет выполнением кода. Другими словами, в ней запускается и поддерживается любой написанный вами код. Традиционно при создании приложений вы пишете код на одном из языков программирования, компилируете его в формат, понятный компьютеру (в единицы и нули), а затем отправляете откомпилированный код на выполнение. Обратите внимание, что компьютеры разных типов говорят на разных языках. Это означает, что для выполнения программы на компьютере другого типа ее надо перекомпилировать. В .NET Framework все обстоит по-другому. Конечно, при работе с .NET Framework надо писать код и заниматься его компиляцией. Однако вместо того чтобы откомпилировать его в то, что понимает компьютер, вы компилируете его в код языка, который называется Microsoft Intermediate Language (MSIL, промежуточный язык корпорации Microsoft). При компиляции в этот код приложение создает так называемые *метаданные*. Они несут в себе информацию, описывающую само приложение. Затем, когда нужно выполнить программу, за дело берется CLR и заново компилирует код — на этот раз на родной язык компьютера. Таким образом, MSIL-код может выполняться на компьютере любого типа, потому что CLR говорит на языках многих компьютеров и делает за вас всю компиляцию на эти языки. Так что откомпилировав свое приложение один раз, его можно передать на любой другой компьютер. Используя метаданные, CLR ищет, каким образом следует запускать приложение. Благодаря этому установка программ становится очень легкой. Дело в том, что при традиционной установке информацию о приложении необходимо помещать в системный реестр, но из-за метаданных системный реестр становится ненужным. Вся необходимая информация хранится вместе с файлами приложения и все ваши изменения вносятся автоматически.

Код, который работает с CLR, называется *управляемым кодом*.

Но это еще не все, что может делать CLR. Он выполняет обработку ошибок, поддерживает безопасность, согласованность версий, а также интеграцию с различными платформами. А это означает, что для написания приложения .NET можно выбрать любой язык с управляемыми кодами, который вы захотите.

Языки программирования .NET

Все языки .NET (Visual C++, Visual C#, Visual J#, JScript и Visual Basic) имеют в своем распоряжении библиотеки классов .NET Framework. Эти библиотеки поддерживают различные технологии (от файлового ввода-вывода и работы

с базами данных до XML) и играют чрезвычайно важную роль в обеспечении межъязыкового взаимодействия приложений, т. к. позволяют разработчикам использовать единый программный интерфейс всех функциональных средств CLR (например, при создании консольных приложений, разработке Windows GUI-приложений (Windows Forms), приложений ASP.NET).

Ваше первое консольное приложение на VB .NET

В качестве примера такой универсальности приведем программы на VB .NET, C# и J# .NET, выводящие сообщение "Hello, World!". Начнем с VB .NET. В текстовом редакторе, например Notepad, создайте файл HelloWorld.vb и в нем наберите код, содержащийся в листинге 1.1. Для компиляции кода VB .NET приложения надо воспользоваться утилитой vbc.exe. В командную строку введите следующую команду:

```
vbc.exe HelloWorld.vb
```

Если в переменной среды `PATH` не прописан путь к каталогу, в котором расположена утилита vbc.exe, то этот путь надо явно указать в команде. Например, в Visual Studio .NET 2003 Final Beta таким путем по умолчанию является `C:\WINDOWS\Microsoft.NET\Framework\v1.1.4322`, следовательно, эта команда примет следующий вид

```
C:\WINDOWS\Microsoft.NET\Framework\v1.1.4322\vbc.exe HelloWorld.vb
```

В результате будет создан файл HelloWorld.exe, который надо запустить на выполнение. Если кому-то покажется неудобным набирать подобную длинную команду (при отладке программы ею придется воспользоваться несколько раз), то создайте BAT-файл, содержащий эту строку, и запускайте его на исполнение или укажите этот путь в системной переменной `PATH`.

Листинг 1.1. Программа Hello, World! на Visual Basic .NET. Файл HelloWorld.vb

```
Imports System
Class HelloWorld
   Shared Sub Main()
      Console.WriteLine("Hello, World!")
   End Sub
End Class
```

Первая инструкция кода (директива `Imports`) импортирует пространство имен `System`, содержащее базовые классы .NET Framework. В частности в нем содержится класс `Console`, инкапсулирующий в себе данные и методы по работе с консольным окном. Из всей совокупности методов этого класса воспользуемся методом `WriteLine`, выводящим строку текста в консольное окно. Весь код в VB .NET, как и в других языках .NET, размещается в классах.

Код класса находится внутри блока `Class … End Class`, у которого есть имя, в данном случае `HelloWorld`. В классе `HelloWorld` имеется единственный метод `Main`, содержащий инструкцию вывода строки текста в консольное окно. Метод `Main` не возвращает значений, поэтому он описывается внутри блока `Sub … Sub Class`. В каждой программе VB .NET должен присутствовать один класс с методом `Main`. Этот метод является входной точкой в программу. Он должен быть помечен модификатором `Shared`, говорящем о том, что метод является статическим.

Параметры утилиты vbc.exe

Утилита vbc.exe имеет целый ряд параметров, управляющих процессом компиляции приложения. В данном разделе перечисляются основные из них:

- `/out:<file>` — задает имя выходного файла;
- `/target:exe` — используется по умолчанию. Создает консольное приложение. Допустима сокращенная версия опции `/t`;
- `/target:winexe` — создает Windows-приложение;
- `/target:library` — создает библиотечный файл;
- `/target:module` — создает модуль, который может быть добавлен в сборку;
- `/addmodule:<file>` — задает ссылку на метаданные из специфицированного модуля;
- `/recurse:<wildcard>` — включает все файлы из данного каталога и его подкаталогов, используя заданный шаблон;
- `/reference:<file_list>` или `/r:` — задает ссылку на метаданные из специфицированного библиотечного файла.

Ваше первое консольное приложение на C#

Сконструируем консольное приложение на C#. В текстовом редакторе, например Notepad, создайте файл HelloWorld.cs и в нем наберите следующий код (листинг 1.2). Для компиляции кода приложения C# надо воспользоваться утилитой csc.exe, поэтому только остается ввести в командную строку следующую команду:

```
csc.exe HelloWorld.cs
```

Листинг 1.2. Программа Hello, World! на Visual C#. Файл HelloWorld.cs

```
using System;

class HelloWorld
{
```

```
    static void Main(string[] args)
    {
        Console.WriteLine("Hello, World!");
    }
}
```

Как легко видеть, код на C# похож на код, написанный на VB .NET. Отличием является то, что в C# в качестве разделителя между операторами используется точка с запятой, а не пробел, как в VB .NET. Кроме того, код группируется посредством открывающей и закрывающей фигурных скобок, а не структур вида *ИмяСтруктуры* `Class` … `End` *ИмяСтруктуры*. Входной точкой в программу также является метод `Main`, он является статическим (модификатор `static`) и не возвращает значений (модификатор `void`). Но у этого метода теперь могут быть параметры, а именно целый массив строк, передаваемых в метод в качестве параметров команды `csc.exe`.

Ваше первое консольное приложение на J#

Сконструируем консольное приложение на J#. В текстовом редакторе, например Notepad, создайте файл HelloWorld.jsl и в нем наберите следующий код (листинг 1.3). Для компиляции кода приложения J# надо воспользоваться утилитой jsc.exe, поэтому остается только ввести в командную строку следующую команду

```
jsc.exe HelloWorld.jsl
```

Листинг 1.3. Программа Hello, World! на Visual J#. Файл HelloWorld.js

```
import System.*;

public class Hello
{
    public static void main(String[] args)
    {
        Console.WriteLine("Hello, World!");
    }
}
```

Пространство имен, директива *Imports*, сборки

Библиотека классов .NET Framework имеет иерархическую структуру с единым корнем и состоит из *пространств имен* (namespaces). Совместно используемые классы в .NET объединяются в структурные единицы, называем

пространством имен. Например, в пространстве имен `System.Drawing` собраны классы, позволяющие работать с графикой, а в пространстве имен `System.IO` — с файлами и каталогами. Пространство имен используется для определения *области видимости* (scope). Два класса с одним и тем же именем могут использоваться в программе при условии, что они принадлежат различным пространствам имен. Объявив пространство имен, разработчик может организовать свое приложение в виде иерархической структуры, основанной на семантически связанных пространствах имен. В формировании одного пространства имен могут участвовать несколько файлов исходного кода. Программист, использующий созданные вами классы, получает доступ к пространству имен через директиву `Imports`

```
Imports [aliasname =] namespace.element
```

- *aliasname* — идентификатор, по которому можно ссылаться внутри модуля на указанное пространство имен (необязательный параметр).
- *namespace* — имя импортируемого пространства имен (обязательный параметр).
- *element* — имя элемента, например класса, пространства имен (необязательный параметр).

В каждом модуле может содержаться произвольное число директив `Imports`, но все они должны располагаться до любой ссылки на идентификаторы. Имя пространства имен является частью полного имени объекта, имеющего в общем случае синтаксис *namespace*.*typename*. Все пространства имен, поставляемые корпорацией Microsoft, начинаются либо с `System` (разработанные командой Microsoft .NET Framework SDK), либо с `Microsoft` (разработанные другими командами Microsoft).

Код в листинге 1.1 можно записать в равносильной форме без использования директивы `Imports` (листинг 1.4).

Листинг 1.4. Консольное приложение Hello, World! без директивы Imports

```
Class HelloWorld HelloWorld
   Sub Main()
      System.Console.WriteLine("Hello, World!")
   End Sub
End Class
```

В данном примере `System` является пространством имен, `Console` — классом, а `WriteLine` — методом класса `Console`.

Над пространствами имен находится другая структурная единица, называемая *сборкой* (assembly), которая является повторно используемым самоописательным строительным блоком приложения, работающего в CLR. Чтобы

в проекте получить доступ к пространствам имен, находящимся в других сборках в этих проектах, надо получить ссылки на соответствующие сборки (см. разд. *"Подключение библиотечного файла в среде Visual Studio"*). Когда ссылка установлена, то, используя полное имя, можно ссылаться на искомый объект.

Например, в следующем коде переменная `p` объявлена как переменная класса `Pen` (перо) пространства имен `System.Drawing`

```
Dim p As System.Drawing.Pen
```

Для сокращения кода в программах применяется директива `Imports`, импортирующая пространства имен. Например, если в начале модуля поместить инструкцию

```
Imports.System.Drawing
```

то предыдущий код можно переписать следующим образом

```
Dim p As Pen
```

Зачем столько языков

Естественно, читатель может задать вопрос: "Зачем столько языков, если все они используют одну и ту же библиотеку классов?" Ответить на него проще всего, сказав, что языки разные нужны, языки разные важны. Если же серьезно, то мир настолько сложен, а задачи, требующие решения, настолько различны, что просто не существует языка, который был бы одинаково удобен и эффективен при решении любой из них. Для одной из задач больше подходит один язык, а для другой — иной. Кроме того, программисты — это обыкновенные люди, и часто они весьма консервативны в своих вкусах и пристрастиях, как и прочие смертные. Как говорится, на вкус и цвет товарищей нет. Поэтому, по всей вероятности, для тех разработчиков, которые любят Visual Basic, корпорация Microsoft разработала Visual Basic .NET, а для тех, кто предпочитает Visual C++ — Visual C++ .NET, для поклонников Java — C# и J#, а JavaScript — Jscript .NET.

Ваше первое консольное приложение, созданное в среде Visual Studio .NET

В заключение главы продемонстрируем, как консольное приложение создается в пакете Visual Studio.NET.

1. Выберите команду **File | New | Project** или нажмите комбинацию клавиш <Ctrl>+<Shift>+<N>. Появится диалоговое окно **New Project**.
2. В левом списке **Project Types** раскройте папку **Visual Basic Projects**, а в правом — выберите значок **Console Application**. Поле **Location** позволяет указать каталог, в котором будет расположен проект, а поле **Name** — задать имя проекта. Введите в это поле имя `HelloWorld` (рис. 1.1).

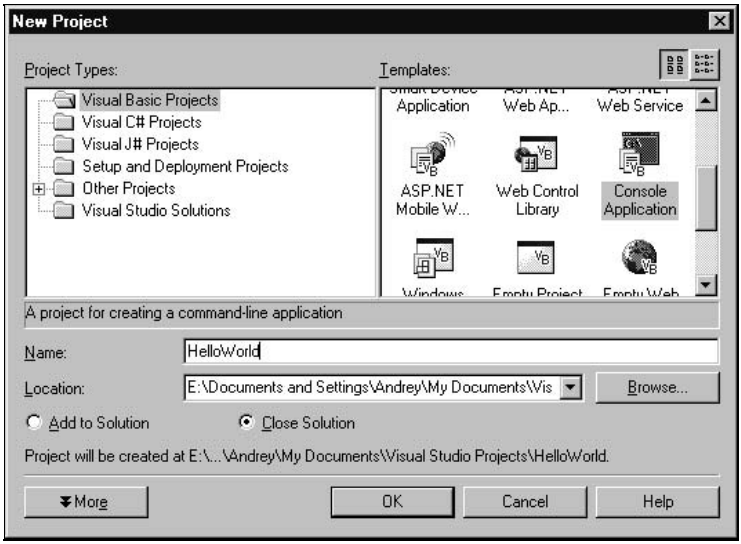


Рис. 1.1. Заполненное окно **New Project**

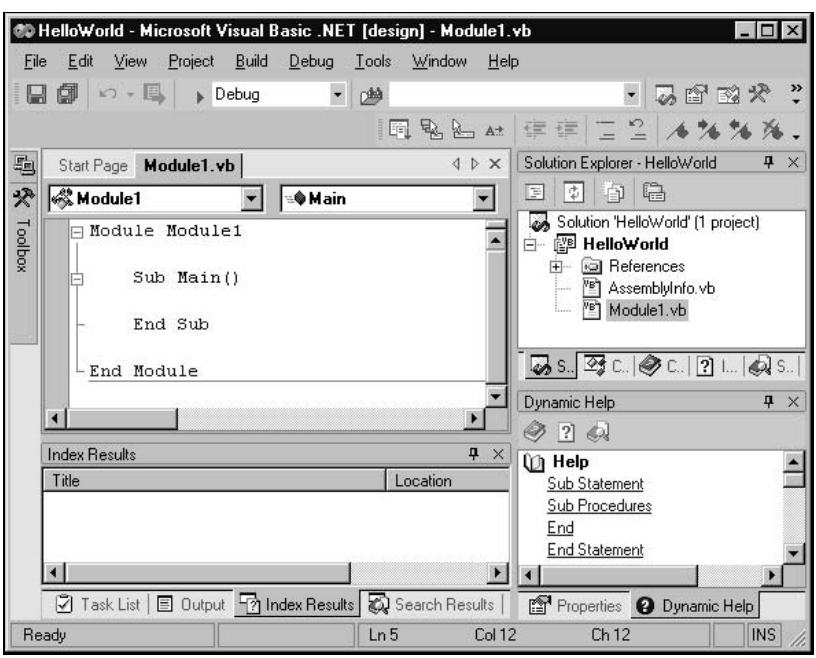


Рис. 1.2. IDE Visual Studio с автоматически сгенерированным кодом

3. Нажмите кнопку **OK**. Диалоговое окно **New Project** закроется. Будет создана заготовка консольного приложения, исполняемый код которого