Алексей Дукин
Антон Пожидаев

Самоучитель Visual Basic 2010

Санкт-Петербург
«БХВ-Петербург»
2010

УДК 681.3.06
ББК 32.973.26-018.2
Д81

Дукин, А. Н.

Д81 Самоучитель Visual Basic 2010 / А. Н. Дукин, А. А. Пожидаев. — СПб.: БХВ-Петербург, 2010. — 560 с.: ил. + Дистрибутив (на DVD)

ISBN 978-5-9775-0512-3

Доступно и подробно описана разработка приложений в среде Visual Basic 2010. Рассмотрены основные понятия объектно-ориентированного программирования и классов, разработка программного интерфейса, работа с файлами, организация печати, методика разработки интернет-приложений, работа с графикой с использованием интерфейса GDI+, создание справочной системы и установочного компакт-диска. Большое внимание уделяется информационным системам, предназначенным для управления базами данных, а также подготовке отчетов с помощью встроенного генератора отчетов. Описаны средства отладки приложений и обработки ошибок. На компакт-диске размещен дистрибутив пакета Microsoft Visual Studio 2010 Express Edition, содержащий Visual Basic 2010 Express Edition и другие компоненты пакета.

Для начинающих программистов

УДК 681.3.06
ББК 32.973.26-018.2

Группа подготовки издания:

Главный редактор	*Екатерина Кондукова*
Зам. главного редактора	*Игорь Шишигин*
Зав. редакцией	*Григорий Добин*
Редактор	*Анна Кузьмина*
Компьютерная верстка	*Ольги Сергиенко*
Корректор	*Зинаида Дмитриева*
Дизайн серии	*Инны Тачиной*
Оформление обложки	*Елены Беляевой*
Зав. производством	*Николай Тверских*

Лицензия ИД № 02429 от 24.07.00. Подписано в печать 31.03.10.
Формат 70×100 1/16. Печать офсетная. Усл. печ. л. 45,15.
Тираж 2000 экз. Заказ №
"БХВ-Петербург", 190005, Санкт-Петербург, Измайловский пр., 29.

Санитарно-эпидемиологическое заключение на продукцию
№ 77.99.60.953.Д.005770.05.09 от 26.05.2009 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 978-5-9775-0512-3

Оглавление

Введение

Обычно Basic ассоциируется с чем-то очень простым в освоении и использовании для программирования. Это действительно так. На заре компьютерных технологий язык BASIC был создан для простых программ и использовался в качестве учебного языка для первых шагов при изучении основ программирования с последующим переходом на более сложные и универсальные языки. Это было заложено в название языка *BASIC — Beginners All-purpose Symbolic Instructional Code*, т. е. многоцелевой код символьных инструкций для начинающих. С прогрессом компьютерных технологий развивался и BASIC. В настоящее время версия Visual Basic 2010 дает возможность решать любые современные задачи разработки приложений. При этом Visual Basic 2010 остался достаточно простым в освоении, став в то же время мощным современным языком программирования.

С помощью Visual Basic 2010 можно создавать приложения практически для любой области современных компьютерных технологий: бизнес-приложения, игры, мультимедиа, базы данных, интернет-приложения. При этом приложения могут быть как простыми, так и очень сложными, в зависимости от поставленной задачи.

Простота и мощность языка Visual Basic 2010 позволили сделать его встроенным языком для приложений Microsoft Office. В настоящее время Visual Basic уже не считается учебным языком — знание Visual Basic и его диалектов (VBA, VBScript) становится необходимостью для современного программиста любого уровня.

В Visual Basic 2010 используются все самые современные методы программирования: объектно-ориентированная модель, включая наследование визуальных классов, модель составных объектов COM (Component Object Model), технология программных компонентов ActiveX. Кроме того, Visual Basic 2010 позволяет создавать многопоточные программы, службы Windows,

разнообразные сетевые приложения и многое другое. Суть этих подходов и их реализацию на примерах можно изучить, прочитав посвященные им главы из этой книги.

Данные методы позволяют опытным программистам быстрее разрабатывать качественное ПО, однако делают разработку менее доступной для начинающих. Поэтому, Microsoft выпустила продукт Small Basic, обладающий рядом преимуществ, которые должны оценить начинающие осваивать программирование:

- очень простая среда разработки — текстовый редактор с многофункциональной подсказкой и лишь несколько кнопок для редактирования текста и запуска программ;
- простой язык, включающий всего 15 ключевых слов;
- встроенная в среду разработки контекстная документация по всем элементам языка;
- возможность расширения компонентов Small Basic для включения дополнительного функционала (такая возможность понравится создателям online-сервисов — можно дать возможность миллионам энтузиастов создать что-то свое с использованием сервиса и Small Basic).

Более подробно о Small Basic можно узнать по ссылке **http://msdn.microsoft.com/ru-ru/devlabs/cc950524.aspx**.

Книга рассчитана на широкий круг пользователей. Начинающему программисту материалы данной книги помогут быстро изучить язык и все основные возможности Visual Basic 2010. Книга будет полезна и читателю, имеющему опыт программирования на предыдущих версиях Visual Basic.

Как построена книга

Книга предполагает последовательное изучение материала от простого к сложному. *Глава 1* является вводной. Вы узнаете, как запустить Visual Basic 2010, как получить в нем справочную информацию, познакомитесь с интегрированной средой разработки, с основными рабочими окнами, а также с настройкой среды разработки программы.

В *главе 2* рассматриваются основы программирования в Visual Basic 2010. Описываются базовые понятия и синтаксические конструкции.

Глава 3 посвящена разработке интерфейса пользователя. Подробно рассмотрен вопрос разработки основ интерфейса: форм и меню. Вы познакомитесь со средствами, предоставляемыми Visual Basic 2010 для создания панелей инструментов, диалоговых окон, строки состояния.

Среда Visual Basic 2010 включает множество элементов управления, позволяющих создать богатый пользовательский интерфейс. В *главе 4* рассматриваются стандартные элементы управления, используемые наиболее часто, такие как надписи, поля различных типов, флажки, списки и т. д. *Глава 5* познакомит вас с размещением в форме графики (в том числе списков с графическими изображениями), полос прокрутки, таймера, объектов, позволяющих работать с датами, добавлением в форму вкладок, облегчающих размещение и группировку большого объема данных, индикатора процесса выполнения, ползунка и гиперссылок.

Язык Visual Basic 2010 является объектно-ориентированным языком. В *главе 6* вы познакомитесь с основными понятиями объектно-ориентированного программирования, такими как инкапсуляция, наследование, полиморфизм. Что такое члены класса, как создавать классы и экземпляры класса, переопределять методы базовых классов и использовать интерфейсы, реализуемые классом, вы также узнаете из этой главы.

При проектировании приложения достаточно часто возникает необходимость в работе непосредственно с файлами. В *главе 7* подробно рассматриваются вопросы добавления, удаления файлов или каталогов, записи данных в файлы, чтения из них данных как программно, так и в интерактивном режиме.

Глава 8 посвящена работе с графикой с использованием интерфейса GDI+. В *главе 9* речь идет о применении в приложениях мультимедиа.

О том, как разработать для своего приложения эффективную справочную систему в формате HTML, вы узнаете из *главы 10*.

Глава 11 посвящена элементам управления данными. Из нее вы узнаете о создании форм для ввода и редактирования данных с использованием компонентов ADO.NET.

Рассмотрению одной из главных задач информационной системы, представлению данных в виде отчетов с помощью генератора отчетов Crystal Reports посвящена *глава 12*. Изучив материал *главы 13*, вы познакомитесь с методикой создания приложений, работающих в Интернете.

Глава 14 посвящена расширенным средствам Visual Basic 2010: созданию системных служб Windows и многопоточному программированию.

Использованию в разработке приложений возможностей других программ, таких, например, как программы, входящие в пакет Microsoft Office, посвящена *глава 15*.

О том, какие средства предоставляет в распоряжение разработчика Visual Basic по отладке приложений и обработке ошибок, вы узнаете из *главы 16*.

Вопросам бесконфликтной работы группы программистов над одним приложением посвящена *глава 17*.

В *главе 18* рассматривается создание инсталлятора, который позволяет устанавливать разработанное вами приложение на компьютере пользователя.

Специальные элементы книги

В книге есть много особых специальным образом выделенных вставок. В них содержится дополнительная информация, облегчающая чтение и поиск информации.

Замечание

В замечаниях речь идет о последствиях, к которым приводят те или иные действия.

Совет

В советах рассказывается о некоторых хитростях, которые следует знать, чтобы наиболее эффективно использовать возможности Visual Basic.

Предупреждение

Предостережения должны помочь вам избежать проблем. В них сказано, чего следует опасаться, а также что нужно делать, чтобы избежать ошибок.

В книге используются различные виды шрифта:

- новые термины выделены *курсивом*;
- команды меню, наименования кнопок, вкладок, опций, флажков, диалоговых окон, областей и т. п. выделены **полужирным шрифтом**;
- две клавиши, соединенные знаком "плюс", — это комбинация клавиш; нажмите первую клавишу и, не отпуская ее, нажмите вторую, а затем отпустите обе;
- названия функций, свойств, методов, баз данных, таблиц, полей таблиц выделены `моноширинным шрифтом`.

При работе с книгой читателю необходимо иметь в виду, что рядом с наименованиями команд меню, диалоговых окон и располагаемых в них элементах даны переводные эквиваленты, а не названия локальной версии.

ГЛАВА 1

Первое знакомство с Visual Basic 2010

Вы приступаете к работе с Visual Basic 2010. Многое из того, с чем вам придется работать (меню, панели инструментов, диалоговые окна), покажутся знакомыми, т. к. они характерны для среды Windows.

Прежде всего, познакомимся с платформой .NET Framework. *.NET Framework* — это вычислительная платформа, которая упрощает разработку приложений в сильно распределенном окружении Интернета. Платформа .NET Framework предлагает следующие возможности:

- предоставление среды выполнения кода, которая уменьшает конфликты между разными версиями языков, гарантирует безопасное выполнение кода, устраняет проблемы, связанные с переносом сред;
- работу с различными типами приложений, размещенных как в Интернете, так и на локальном компьютере;
- установление стандартов для поддержки платформы .NET Framework другими языками;
- создание непротиворечивой объектно-ориентированной среды программирования, в которой код объекта может храниться и выполняться локально, выполняться локально, но быть распределенным в Интернете или выполняться удаленно.

Платформа .NET Framework состоит из двух частей: *единой среды исполнения* (Common Language Runtime, CLR) и *библиотеки классов*. Единая среда исполнения управляет кодом во время его выполнения, обеспечивая управление памятью, потоком и удаленное управление и гарантируя безопасность. Код, генерируемый CLR, называется *управляемым кодом*, а код, не использующий возможности CLR, — неуправляемым. Библиотека классов является всесторонней, объектно-ориентированной коллекцией типов, которую можно ис-

пользовать для разработки приложений, начиная с традиционных приложений с командной строкой и с использованием графического пользовательского интерфейса и заканчивая приложениями, использующими Web-формы и XML Web-сервисы.

Запуск Visual Basic

Для запуска программы из главного меню Windows выполните следующие действия:

1. Нажмите кнопку **Пуск**, расположенную в нижней части экрана.
2. В открывшемся главном меню Windows выберите команду **Программы**. Появится меню данной команды.
3. В открывшемся главном меню выберите в меню команду **Microsoft Visual Studio 2010**. Появится меню данной команды.
4. Выберите в меню команду **Microsoft Visual Studio 2010**. На экране появится главное окно программы (рис. 1.1).

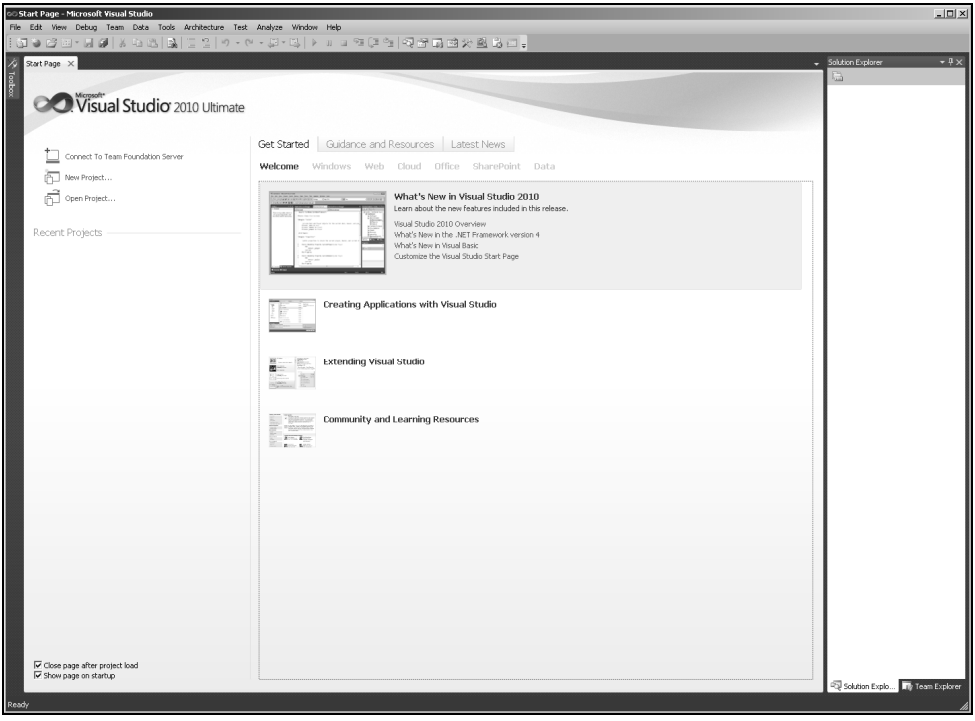

Рис. 1.1. Главное окно Microsoft Visual Studio 2010

Совет

Для более быстрого запуска программы можно создать на рабочем столе ярлык и назначить ему клавиши быстрого вызова. Тогда будет достаточно нажать заданную комбинацию клавиш для запуска программы. Кроме того, для удобства запуска приложения можно использовать панель **Быстрый запуск** системы Windows.

Главное окно

На рис. 1.1 показано главное окно Visual Studio 2010, каким оно выглядит после запуска программы.

В нем можно выделить несколько основных объектов: стандартная панель инструментов, окна **Start Page** (Начальная страница), **Solution Explorer** (Обозреватель решений), **Toolbox** (Инструментарий).

Visual Studio 2010 предоставляет в распоряжение пользователя набор самых разнообразных *панелей инструментов*. Эти панели инструментов содержат кнопки, назначение которых зависит от функций конкретной панели инструментов. После запуска Visual Studio 2010 на экране отображается *стандартная панель инструментов*.

Диалоговое окно **Start Page** (Начальная страница) позволяет открывать недавно использовавшиеся проекты, осуществляет поиск примеров, как из справочной системы, так и Интернета, содержит различные ссылки на сайты, которые могут помочь при работе с Visual Studio 2010.

В окне **Solution Explorer** (Обозреватель решений) размещаются проекты и файлы текущего решения.

Для получения подробной информации об объектах используется диалоговое окно **Object Browser** (Просмотр объектов). Оно позволяет искать и исследовать элементы, их свойства, методы, события, находящиеся в проектах и ссылках на них, как бы представляя собой внутреннюю библиотеку.

Для удобства разработки существует окно **Toolbox** (Инструментарий), отображающее элементы, используемые в проектах Visual Basic. К таким элементам могут относиться компоненты .NET и COM, HTML-объекты, фрагменты кода, текст.

В главном диалоговом окне **Visual Basic** могут отображаться и другие окна, но они будут рассмотрены позднее.

Создание нового проекта

Для создания нового проекта используется диалоговое окно **New Project** (Новый проект) (рис. 1.2).

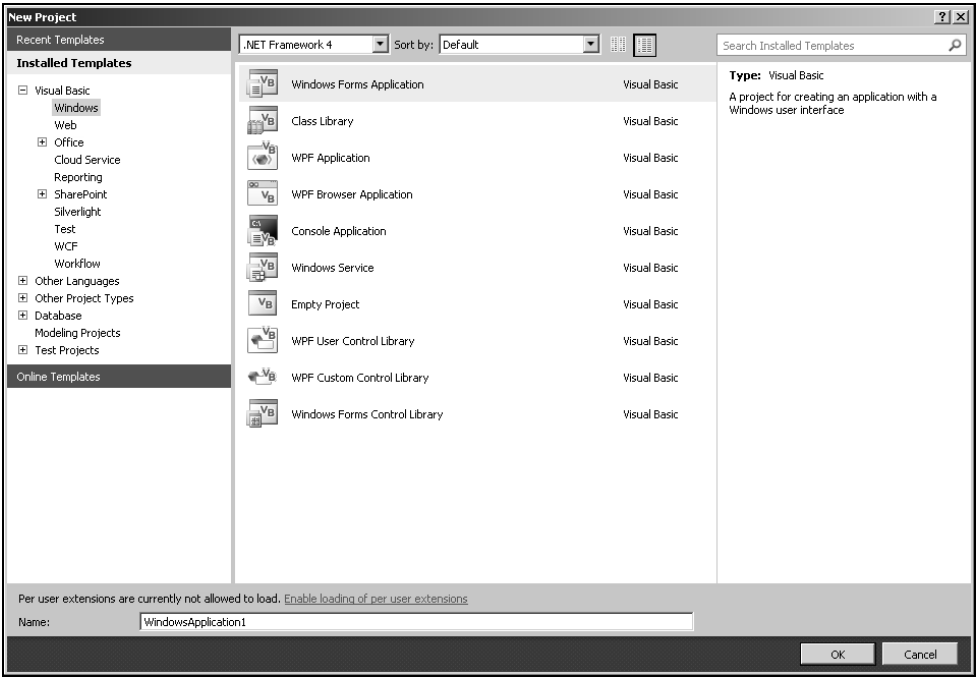

Рис. 1.2. Диалоговое окно **New Project**

Для его открытия выполните одно из следующих действий:

- в окне **Start Page** (Начальная страница) выберите ссылку **Create Project** (Создать проект);
- из меню **File** (Файл) выберите пункт **New Project** (Новый проект);
- нажмите кнопку **New Project** (Новый проект)  , расположенную на стандартной панели инструментов.

Visual Basic 2010 предлагает создание различных приложений, каждое из которых включает собственный проект и шаблоны для упрощения работы. Остановимся на основных проектах.

- **Class Library** (Библиотека классов). Создает класс или компонент, который может использоваться в других приложениях. Этот тип проекта не предполагает создания форм.
- **Console Application** (Консольное приложение). Создает приложения, которые, как правило, не поддерживают пользовательские графические интерфейсы, и представляет собой отдельный исполняемый файл.
- **Empty Project** (Пустой проект). Создает пустой проект, содержащий только необходимую для хранения приложения структуру файлов. Какие-либо ссылки, компоненты и файлы нужно добавлять вручную.

- **Windows Forms Application** (Windows-приложение). Предназначен для создания традиционного Windows-приложения и клиент-серверного приложения, пользовательский интерфейс для которых проектируется с помощью форм Windows. Для формы можно задавать определенные характеристики и располагать на ней различные элементы управления.
- **Windows Forms Control Library** (Библиотека элементов управления Windows). Является аналогом ActiveX Controls в Visual Basic 6. Позволяет создавать индивидуальные средства управления для форм Windows.
- **Windows Service** (Сервис Windows). Создает сервисы Windows, более известные как NT-сервисы, способные создавать исполнимые приложения. Эти сервисы могут автоматически запускаться при загрузке компьютера, быть приостановлены и перезапущены. Они не имеют пользовательских интерфейсов, а информация выводится в файл протокола. Эти сервисы идеальны для серверов, которые требуют длительной работы и не взаимодействуют с другими пользователями, работающими на том же компьютере.
- **WPF Application** (WPF-приложение). Открывает шаблон для создания клиентского WPF-приложения.
- **WPF Browser Application** (Браузерное WPF-приложение). Открывает шаблон для создания WPF для Web-приложений.
- **WPF Custom Control Library**, **WPF User Control Library.** Создают класс или компонент, который может использоваться в других WPF-приложениях. Эти типы проекта не предполагают создания форм.

Замечание

WPF (Windows Presentation Foundation) — подсистема интерфейса пользователя и программный интерфейс на основе XML и векторной графики в составе .NET Framework 3.0 и выше. WPF представляет собой высокоуровневый объектно-ориентированный функциональный слой (framework), позволяющий создавать 2D- и 3D-интерфейсы.

Для построения отчетов используются проекты из пункта **Reporting**:

- **Crystal Reports Application** (Приложение с использованием Crystal Reports). Открывает шаблон для создания отчетов с помощью Crystal Reports.
- **Reports Application** (Приложение для отчетов). Открывает шаблон для создания отчетов с пользовательским интерфейсом Windows.

Для интерактивного, или иначе онлайнового, поиска шаблонов необходимо в окне **New Project** (Новый проект) выбрать пункт **Online Templates** (Шаблоны в Интернете).

Главное меню

При работе с Visual Basic 2010 можно использовать как кнопки панели инструментов, так и строку меню, расположенную в верхней части экрана, все команды которого являются иерархическими. При выборе определенной команды открывается ее подменю.

Главное меню может содержать следующие пункты: **File** (Файл), **Edit** (Правка), **View** (Вид), **Project** (Проект), **Build** (Сборка), **Debug** (Отладка), **Data** (Данные), **Format** (Формат), **Tools** (Сервис), **Test** (Тестирование), **Window** (Окно) и **Help** (Справка). Первоначально при запуске программы в меню присутствуют лишь некоторые из указанных выше пунктов. Они добавляются в меню при открытии дополнительных окон. Например, при открытии проекта в меню добавляются пункты **Project** (Проект), **Build** (Сборка), **Data** (Данные) и **Debug** (Отладка).

Меню *File*

Меню **File** (Файл) содержит команды, связанные с доступом к файлам. Эти команды позволяют создавать новые файлы и проекты, открывать существующие, закрывать, сохранять и печатать их. Также можно добавлять новые и существующие проекты и элементы (формы, классы, файлы и т. д.) в решение.

В нижней части меню располагаются имена последних открываемых файлов и проектов, которые предоставляют возможность быстрого открытия любого из них. Последней командой меню **File** (Файл) является команда **Exit** (Выход), которая предназначена для выхода из Visual Basic.

Замечание

Первоначально в меню может отображаться лишь часть существующих команд. Настроить отображение пунктов меню можно с помощью диалогового окна **Customize** (Настроить), открываемое с помощью пункта меню **Customize** (Настроить) меню **Tools** (Сервис).

Меню *Edit*

Меню **Edit** (Правка) имеется во многих приложениях Windows. Команды этого меню используются при создании форм, редактировании программ.

Данное меню содержит стандартные команды редактирования: отменить предыдущую команду, вернуть предыдущую команду, вставить, вырезать фрагмент текста, выделить весь текст, удалить выделенное, а также команды поиска и замены фрагмента текста и перемещение к определенной строке.

С помощью команд пункта **IntelliSense** меню **Edit** (Правка) можно просмотреть список членов определенного класса, структуры, объединения или пространства имен и вставить в код подходящий, получить информацию о числе, типах и именах параметров методов и свойств, дописать слова, являющиеся именами методов, команд.

Команда **Insert Snippet** (Вставить фрагмент кода) пункта **IntelliSense** меню **Edit** (Правка) отображает иерархический список наиболее часто выполняемых задач. При выборе одного из пунктов списка в программу добавляется фрагмент кода, позволяющий выполнять указанную задачу.

Команды пункта **Outlining** (Скрытие) позволяют скрыть часть кода программы с помощью символа  , отменить использование этого символа, раскрыть весь код программы. Нажав на плюс рядом с этим символом, можно просмотреть скрытый код.

В данном меню также могут содержаться команды, которые позволяют выделять текст как комментарий, показывать непечатаемые символы, устанавливать перенос слов, делать все выделенные символы строчными или прописными и т. д.

Меню *View*

Данный пункт меню содержит команды вызова окон среды Visual Basic. С помощью этих команд могут открываться окна редактора программного кода, конструктора формы, свойств объектов, обозревателя решений и другие окна.

Команда **Toolbars** (Панели инструментов) позволяет управлять отображением всевозможных панелей инструментов.

Внизу меню **View** (Вид) располагается команда **Property Pages** (Страницы свойств), открывающая окно для определения общих настроек текущего проекта.

Меню *Project*

Меню **Project** (Проект) содержит команды, позволяющие добавлять в проект и удалять из него элементы проекта, такие как форма, программный модуль, класс, а также дающие возможность добавлять ссылки на подключаемые библиотеки.

Последней командой меню **Project** (Проект) является команда **Properties** (Свойства), позволяющая открыть окно свойств проекта.

Меню *Build*

Меню **Build** (Сборка) содержит команды, помогающие скомпоновать решение или проект.

Меню *Debug*

В меню **Debug** (Отладка) расположены команды, предназначенные для отладки и запуска приложения. С помощью команд этого меню можно запустить приложение на выполнение, установить точки останова программы, осуществить пошаговое выполнение приложения, открыть специальные окна для отладки.

Меню *Format*

Этот пункт меню доступен при работе в конструкторе форм. Меню **Format** (Формат) содержит команды, управляющие выравниванием текста и объектов, заданием размеров объектов и определением интервалов между ними. Однако при работе с различными конструкторами становятся доступными и дополнительные команды.

Внизу этого меню располагается команда **Lock Controls** (Блокировка элементов управления), которая позволяет сделать недоступным перемещение элементов управления и сохранить их размер. С ее помощью фиксируются все элементы управления, включая саму форму.

Меню *Tools*

Меню **Tools** (Сервис) содержит средства для настройки среды разработки, создания макросов, а также команды запуска дополнительных утилит.

Меню *Window*

В меню **Window** (Окно) располагаются команды, которые управляют открытыми на экране окнами. С помощью этих команд можно упорядочивать, скрывать окна и переходить из одного окна в другое. Кроме того, команды данного меню позволяют активизировать любое открытое окно.

Меню *Help*

Help (Справка) — последняя команда меню главного окна. Используя команды данного меню, можно вызвать справочную систему с различными вариантами представления информации, а также получить сведения о системе.

Стандартная панель инструментов

В Visual Basic Express Edition содержится большое количество панелей инструментов для отладки и запуска программ, задания расположения элементов на форме и многого другого. Познакомимся со стандартной панелью инструментов (рис. 1.3), которая используется во всех режимах работы.

Рис. 1.3. Стандартная панель инструментов

Назначение кнопок стандартной панели инструментов описано в табл. 1.1.

Таблица 1.1. *Назначение кнопок стандартной панели инструментов*

Кнопка	Название	Назначение
	**New Project** (Новый проект)	Создает новый проект
	**New Web Site** (Новый сайт)	Позволяет создать новый сайт
	**Open File** (Открыть файл)	Открывает существующий файл
	**Add New Item** (Добавить новый элемент)	Позволяет добавить в проект новый или существующий элемент, форму, модуль, класс, компонент или элемент управления
	**Save** (Сохранить)	Сохраняет открытый файл
	**Save All** (Сохранить все)	Сохраняет все открытые файлы
	**Cut** (Вырезать)	Удаляет выделенный текст или выделенные объекты и помещает их в буфер
	**Copy** (Копировать)	Копирует в буфер выделенный текст или выделенные объекты, не удаляя их
	**Paste** (Вставить)	Вставляет содержимое буфера
	**Find** (Найти)	Осуществляет поиск информации по контексту
	**Comment out the selected lines** (Закомментировать выделенные строки)	Комментирует выделенные строки кода
	**Uncomment the selected lines** (Снять комментарий выделенных строк)	Убирает комментирование выделенных строк кода
	**Undo** (Отменить)	Отменяет выполненные действия

Таблица 1.1 *(продолжение)*

Кнопка	Название	Назначение
	**Redo** (Восстановить)	Восстанавливает отмененные действия
	**Navigate Backward** (Назад)	Возвращает на предыдущую вкладку
	**Navigate Forward** (Вперед)	Возвращает на вкладку, которая была до перехода на предыдущую
	**Start Debugging** (Запустить отладку)	Запускает программу на выполнение
	**Break All** (Остановить)	Приостанавливает процесс отладки приложения
	**Stop Debugging** (Остановить отладку)	Прекращает выполнение программы
	**Step Into** (Вход в процедуру)	Выполняет программу по шагам с заходом в подпрограммы
	**Step Over** (Перешагивание)	Выполняет программу по шагам без захода в подпрограммы
	**Step Out** (Выход из процедуры)	Выполняет программу до возврата из текущей функции
	**Solution Explorer** (Обозреватель решения)	Открывает окно обозревателя решений
	**Properties Window** (Окно свойств)	Открывает окно **Properties** (Свойства), используемое для настройки свойств проекта, файлов и их элементов
	**Team Explorer** (Обозреватель команд)	Открывает окно **Team Explorer** (Обозреватель команд), отображающее сервер и доступные проекты на Team Foundation Server (Сервер командной разработки)
	**Object Browser** (Обзор объектов)	Открывает окно **Object Browser** (Просмотр объектов), позволяющее просматривать классы, свойства, методы, события и константы выбранных библиотек
	**Toolbox** (Инструментарий)	Открывает окно **Toolbox** (Инструментарий), содержащее элементы, используемые разработчиком при создании приложения
	**Extension Manager** (Менеджер расширений)	Открывает окно **Extension Manager** (Менеджер расширений), позволяющее получать доступ к расширениям и загружать их прямо из IDE
	**Error List** (Список ошибок)	Открывает окно **Error List** (Список ошибок), содержащее список ошибок, предупреждений и замечаний по приложению

Таблица 1.1 *(окончание)*

Кнопка	Название	Назначение
	**Immediate** (Непосредственное выполнение)	Открывает окно **Immediate Window** (Окно непосредственного выполнения), предназначенное для ручного ввода и выполнения команд

Замечание

Если по внешнему виду кнопки вы не можете определить ее назначение, наведите курсор мыши на кнопку. При этом под курсором появится всплывающая подсказка с ее наименованием.

По умолчанию в главном окне программы Visual Basic всегда присутствует стандартная панель инструментов, если только вы не удалили ее с экрана. Если для работы необходима стандартная панель инструментов, выберите в подменю **Toolbars** (Панели инструментов) меню **View** (Вид) команду **Standard** (Стандартная).

После установки стандартная панель инструментов размещается в верхней части главного окна, но она, как и все остальные панели инструментов, может перемещаться в любое место экрана. Для этого установите курсор мыши в любое свободное от кнопок место на панели инструментов, нажмите кнопку мыши и, не отпуская ее, переместите панель на новое место.

Окно *Start Page*

Окно **Start Page** (Начальная страница) (рис. 1.4) позволяет просмотреть последние использовавшиеся проекты, а также просмотреть ссылки на сайты, содержащие новости о продукте Visual Studio, документацию, учебные пособия.

Начальная страница автоматически открывается при запуске Visual Studio 2008. Если же окно **Start Page** (Начальная страница) не появилось, его можно вызвать с помощью команды **Start Page** (Начальная страница) меню **View** (Вид).

Диалоговое окно **Start Page** (Начальная страница) содержит следующие разделы:

- **Recent Projects** (Последние проекты) — список последних открываемых проектов, а также кнопки **Connect To Team Foundation Server** (Соединение с сервером командной разработки), **New Project** (Создать проект) и **Open Project** (Открыть проект);
- **Get Started** (Для начала) — ссылки на темы справочной системы, которые позволяют получить начальные сведения для работы с Visual Basic;

- **Guidance and Resources** (Руководства и ресурсы) — ссылки на более общие руководства по кодированию и разработке;
- **Latest News** (Последние новости) — обновленные по RSS статьи о новых технологиях и продуктах компании Microsoft.

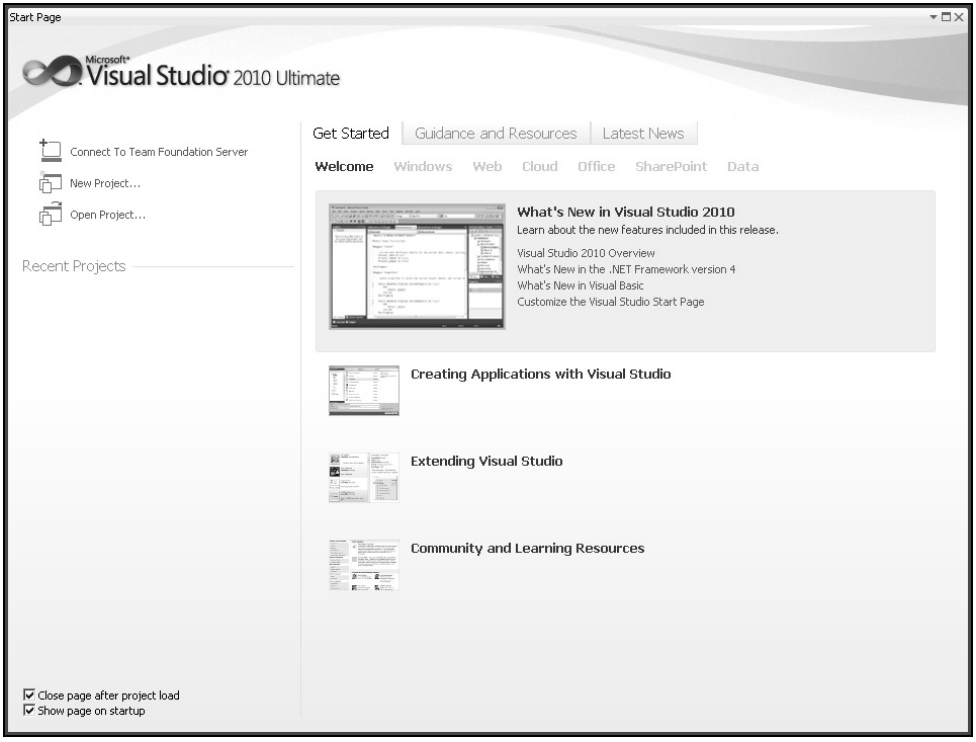

Рис. 1.4. Диалоговое окно **Start Page**

Окно конструктора форм

Окно конструктора форм является основным рабочим окном, в котором выполняется визуальное проектирование приложения (рис. 1.5). Вызвать это окно можно из главного меню командой **Designer** (Конструктор) меню **View** (Вид) или двойным щелчком на названии формы в окне обозревателя решений.

В окне конструктора форм визуально создаются все формы приложения с использованием инструментария среды разработки. Для точного позиционирования объектов на форме в окне можно использовать сетку. Размер ячеек сетки можно изменять. Для определения задаваемых по умолчанию парамет-

ров сетки предназначена вкладка **Windows Forms Designer** (Конструктор форм) диалогового окна **Options** (Параметры), открываемого командой **Options** (Параметры) из меню **Tools** (Сервис).

Размер формы в окне можно изменять, используя маркеры выделения формы и мышь. Для изменения размера формы необходимо установить указатель мыши на маркер и, когда он примет вид двунаправленной стрелки, перемещать до получения требуемого размера.

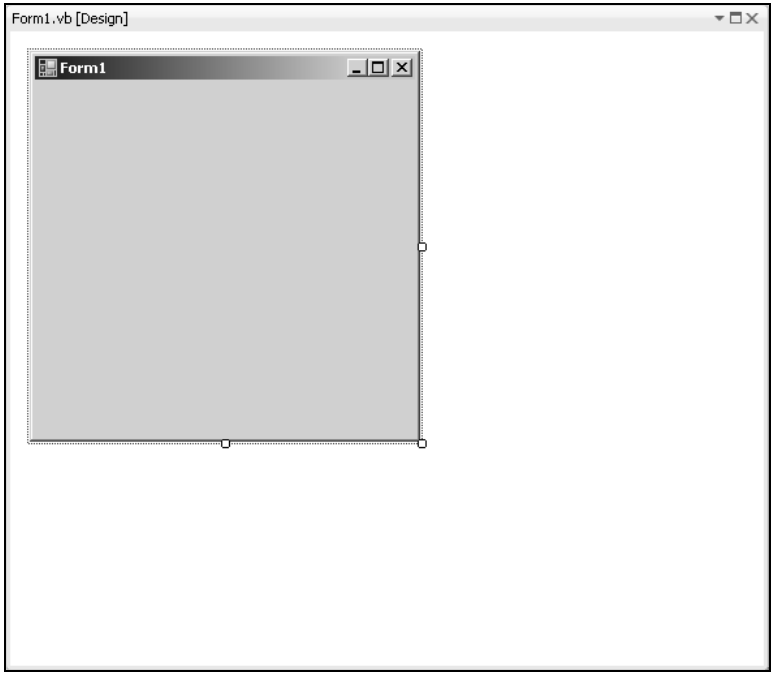

Рис. 1.5. Окно конструктора форм Visual Basic 2010

Окно редактора кода

Редактор кода программы — это мощный текстовый редактор с большим количеством возможностей, являющийся основным инструментом программиста для создания и отладки приложения.

В окне редактора кода (рис. 1.6) расположены следующие элементы:

- раскрывающийся список **Class Name** (Имя класса) содержит перечень объектов приложения. Этот список находится в левом верхнем углу окна редактора. При выборе объекта в этом списке содержимое списка **Method Name** (Имя метода) изменяется;

- раскрывающийся список **Method Name** (Имя метода) дает возможность выбора членов объекта (событий) и автоматического вывода в окно редактора процедуры или шаблона для выбранного члена. Данный список располагается справа от списка **Class Name** (Имя класса).

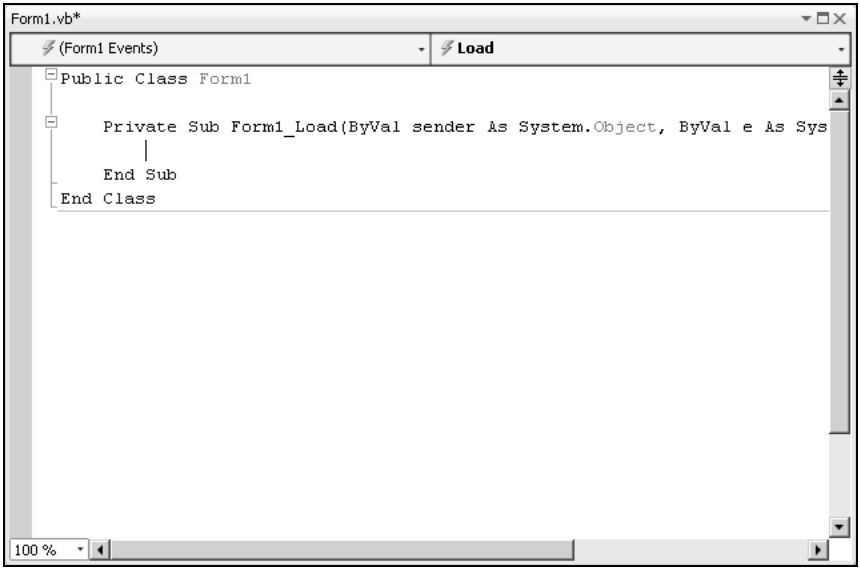

Рис. 1.6. Окно редактора кода

Редактор кода вызывается командой **View Code** (Открыть код) контекстного меню формы или командой **Code** (Код) меню **View** (Вид). Для каждого элемента проекта (формы, программного модуля) открывается отдельная вкладка в окне редактора кода. Для работы в окне редактора можно использовать контекстное меню, которое содержит указанные в табл. 1.2 команды.

Таблица 1.2. *Команды контекстного меню редактора кода*

Команда	Назначение
Insert Snippet (Вставить фрагмент кода)	Позволяет добавить фрагмент кода из списка примеров
Go To Definition (Перейти к описанию)	Переходит к описанию указанной функции, переменной или константы
Insert Breakpoint (Вставить точку останова)	Вставляет точку останова
Run To Cursor (Выполнить до курсора)	Позволяет выполнить программу от текущей выполняемой строки до строки с установленным в ней текстовым курсором

Таблица 1.2 *(окончание)*

Команда	Назначение
Cut (Вырезать)	Вырезает выделенный текст в буфер обмена
Copy (Копировать)	Копирует выделенный текст в буфер обмена
Paste (Вставить)	Вставляет текст из буфера обмена

Окно *Solution Explorer*

Для того чтобы получить доступ к компонентам, входящим в решение, используется диалоговое окно **Solution Explorer** (Обозреватель решений). Если после создания или открытия решения этого окна нет на экране, то можно его отобразить одним из следующих способов:

- в меню **View** (Вид) выберите команду **Solution Explorer** (Обозреватель решений);
- нажмите кнопку **Solution Explorer** (Обозреватель решений) стандартной панели инструментов;
- нажмите комбинацию клавиш <Ctrl>+<Alt>+<L>.

Откроется окно обозревателя решений, содержащее список всех компонентов решения. В верхней части окна размещается имя решения, а ниже располагаются входящие в него проекты и файлы. На рис. 1.7 в решение входят два проекта. Видно, что имя второго проекта выделено. Это говорит о том, что при запуске приложения именно данный проект будет запускаться.

Окно обозревателя решений содержит кнопки (табл. 1.3), отображение которых зависит от типа выделенного в окне компонента.

Рис. 1.7. Окно **Solution Explorer**

Таблица 1.3. *Кнопки диалогового окна* ***Solution Explorer***

Кнопка	Название	Описание
	**Properties** (Свойства)	Открывает окно свойств для указанного объекта
	**Show All Files** (Показать все файлы)	Позволяет отобразить все файла проекта, в том числе исключенные и скрытые
	**Refresh** (Обновить)	Обновляет содержимое обозревателя решений
	**View Code** (Открыть код)	Открывает окно редактора кода для указанного объекта
	**View Designer** (Открыть конструктор)	Открывает окно конструктора формы для указанной формы
	**View Class Diagram** (Открыть диаграмму классов)	Открывает окно диаграммы классов для указанной формы

Окно *Toolbox*

Панель элементов управления — основной рабочий инструмент при визуальной разработке форм приложения (рис. 1.8). Панель элементов управления вызывается из меню **View** (Вид) командой **Toolbox** (Инструментарий) или нажатием кнопки **Toolbox** (Инструментарий) на стандартной панели инструментов.

Панель элементов управления состоит из различных разделов, в которых расположены используемые в проектах элементы. В табл. 1.4 описаны основные разделы окна **Toolbox** (Инструментарий).

Рис. 1.8. Окно **Toolbox**

Таблица 1.4. *Разделы окна* ***Toolbox***

Раздел	Описание
All Windows Forms (Все формы)	В этом разделе располагаются все элементы управления формы из указанных ниже разделов
Common Controls (Основные элементы управления)	Хранит основные элементы управления, используемые для построения интерфейса пользователя
Containers (Контейнеры)	В этом разделе хранятся элементы, которые могут содержать другие объекты. Например, вкладка и панель
Menus & Toolbars (Меню и панели задач)	Содержит такие элементы управления, как обычное и контекстное меню, строка состояния и панель инструментов
Data (Данные)	Содержит компоненты, с помощью которых можно получить доступ к данным и источникам данных
Components (Компоненты)	Хранит элементы, посредством которых можно выполнять мониторинг файловой системы, запись информации об ошибках, возникающих при выполнении приложения, и т. д.
Printing (Печать)	В этом разделе содержатся элементы, которые используются для организации печати
Dialogs (Диалоговые окна)	Содержит список стандартных диалоговых окон: окна открытия и сохранения файла, настройки шрифтов текста и цветовой палитры
WPF Interoperability (WPF-взаимодействие)	Содержит элементы, используемые в WPF
Reporting (Отчеты)	Содержит элементы, используемые для создания отчетов
Visual Basic PowerPacks	Содержит дополнительные элементы управления Windows Forms
General (Общие)	В этом разделе могут располагаться стандартные элементы управления проекта и специальные управляющие элементы

Замечание

В окне **Toolbox** (Инструментарий) всегда присутствует раздел **General** (Общие). При открытии редакторов и конструкторов добавляются другие разделы. Кроме того, можно создать собственный раздел.

Окно *Properties*

Окно **Properties** (Свойства) предназначено для отображения и настройки свойств объектов решения, включая форму и размещенные в ней объекты. В нем, например, содержатся такие свойства выбранного объекта, как позиция в форме, высота, ширина, цвет (рис. 1.9).

Для открытия диалогового окна **Properties** (Свойства) выполните одно из следующих действий:

- в меню **View** (Вид) выберите команду **Properties Window** (Окно свойств);
- нажмите кнопку **Properties Window** (Окно свойств)  , расположенную на стандартной панели инструментов;
- выберите команду **Properties** (Свойства) контекстного меню выделенного объекта;
- нажмите клавишу <F4>.

Поскольку форма и элементы управления каждый сам по себе является объектом, то набор свойств в этом окне меняется в зависимости от выбранного объекта. С помощью кнопок **Alphabetical** (По алфавиту)  и **Categorized** (По категориям)  свойства объекта можно просмотреть в алфавитном порядке или по группам (категориям) соответственно.

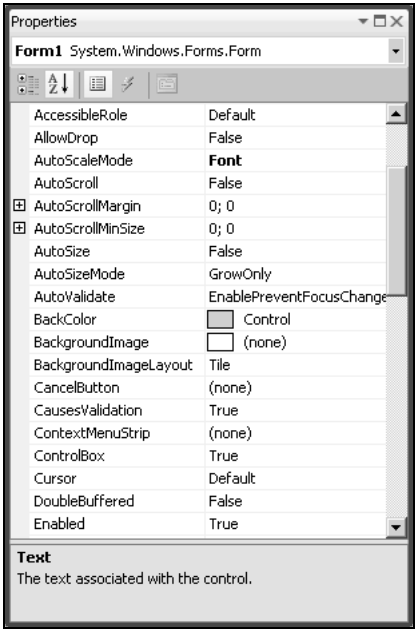

Рис. 1.9. Окно свойств объектов **Properties**

В нижней части окна появляется подсказка, поясняющая назначение выбранного свойства объекта. Более подробное пояснение можно посмотреть в справочной системе Visual Basic, выделив интересующее свойство и нажав клавишу <F1>. Используя диалоговое окно **Properties** (Свойства), можно из-

менить установленные по умолчанию свойства объектов. Часть свойств объекта, например размеры и расположение, можно задать перемещением объекта и изменением его размеров с помощью мыши в конструкторе форм. Свойства, установленные в окне свойств, можно изменять при выполнении приложения, написав соответствующие коды в процедурах, создаваемых с помощью редактора кода.

Как правило, форма содержит много объектов. Если выбрать сразу несколько объектов, то в окне свойств можно увидеть общие для этих объектов свойства.

Подробнее окно свойств будет описано в *главе 3*.

Окно *Object Browser*

Для просмотра всех элементов, входящих в состав решения, предназначено окно **Object Browser** (Просмотр объектов). В этом окне (рис. 1.10) можно получить доступ не только ко всем входящим в решение элементам, но и их свойствам, методам, событиям. Окно просмотра объектов обычно не визуализировано, и его можно вызвать командой **Object Browser** (Просмотр объектов) из меню **View** (Вид) или нажатием кнопки **Object Browser** (Просмотр объектов)  , расположенной на стандартной панели инструментов.

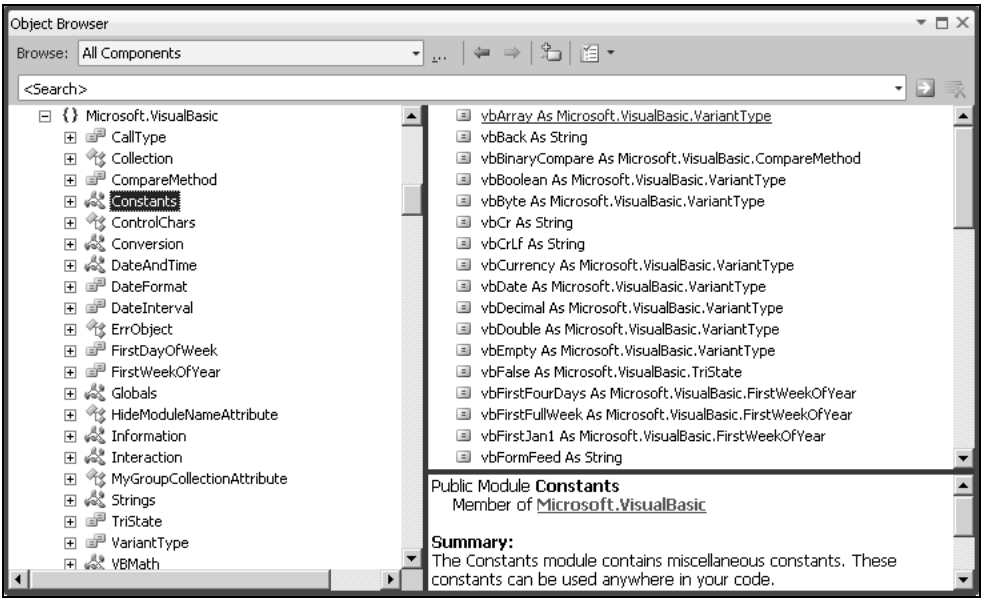

Рис. 1.10. Окно просмотра объектов **Object Browser** Visual Basic 2010

С помощью раскрывающегося списка **Browse** (Просмотреть) окна **Object Browser** (Просмотр объектов) задается область просмотра. Этот список может содержать следующие значения:

- **All Components** (Все компоненты) — в окне **Object Browser** (Просмотр объектов) будут отображаться все имеющиеся компоненты;
- **.NET Framework** (Компоненты платформы .NET Framework) — окно просмотра объектов будет содержать все библиотеки классов платформы .NET Framework;
- **Silverlight 3.0** (Компоненты Silverlight) — окно просмотра объектов будет содержать все библиотеки классов Silverlight;
- **My Solution** (Компоненты открытого решения) — в окне **Object Browser** (Просмотр объектов) будут отображаться содержащиеся в открытом решении пространства имен, классы, структуры, интерфейсы, типы и их свойства, методы, события, переменные, константы;
- **Custom Component Set** (Список выбранных пользователем компонентов) — в окне просмотра объектов будет отображаться содержимое компонентов, указанных в диалоговом окне **Edit Custom Component Set** (Редактировать список выбранных пользователем компонентов), которое открывается с помощью расположенной справа от списка кнопки **Add Other Components** (Добавить другие компоненты). В качестве компонентов могут выступать проекты решения, COM-объекты, внешние библиотеки и исполняемые файлы.

Для задания типа отображаемых компонентов предназначена кнопка **Object Browser Settings** (Настройки окна просмотра объектов)  , при нажатии которой открывается меню со списком типов компонентов: пространства имен, контейнеры, базовые, производные и скрытые типы, открытые, защищенные и скрытые члены классов. При выборе типа слева от его наименования появляется галочка.

Окно *Locals*

Окно **Locals** (Локальные переменные) служит для просмотра списка локальных переменных приложения и контроля их значений (рис. 1.11). Вызвать его можно командой **Locals** (Локальные переменные) из подменю **Windows** (Окна) меню **Debug** (Отладка).

В окне **Locals** (Локальные переменные) удобно просматривать имена локальных переменных, объявленных в текущей процедуре, тип и значения этих переменных. Указанная информация автоматически появляется в окне при

его вызове. Данное окно используется для отладки и проверки работы приложений. С его помощью можно проконтролировать все локальные переменные в точках останова программы.

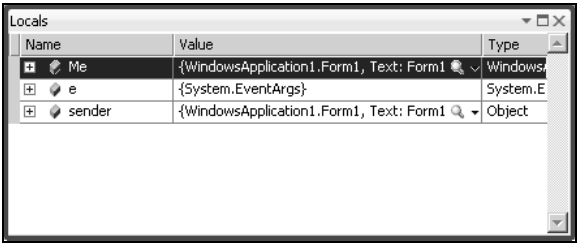

Рис. 1.11. Окно **Locals**

Замечание

При работе с окном **Locals** (Локальные переменные) необходимо иметь в виду, что глобальные переменные в нем для просмотра недоступны.

Если окно открыто постоянно, то данные между точками остановки программы при работе приложения автоматически обновляются.

В окне **Locals** (Локальные переменные) можно не только просматривать переменные и их значения в данный момент работы программы. Очень полезным свойством этого окна является возможность изменять значения переменных для проверки реакции программы на заданные значения. Для этого в столбце **Value** (Значение) необходимо щелкнуть мышью на изменяемом значении. При этом значение перейдет в режим редактирования, и его можно будет изменить. Клавишей <Enter> или перемещением указателя мыши на другое поле устанавливается новое значение, если оно имеет допустимое значение.

Окно *Immediate Window*

Окно **Immediate Window** (Окно непосредственного выполнения) предназначено для ручного ввода и выполнения команд (рис. 1.12). Вызвать его можно командой **Immediate** (Непосредственное выполнение) из подменю **Windows** (Окна) меню **Debug** (Отладка).

Данное окно удобно для отладки и проверки работы программы в пошаговом режиме, а также при необходимости протестировать созданные методы или иные фрагменты написанного кода в режиме редактирования. Проверяемые части или блоки программы можно копировать из программных модулей приложения в окно **Immediate Window** (Окно непосредственного выполне-

ния) и после проверки и внесения необходимых изменений по результатам контроля возвращать в модуль приложения.

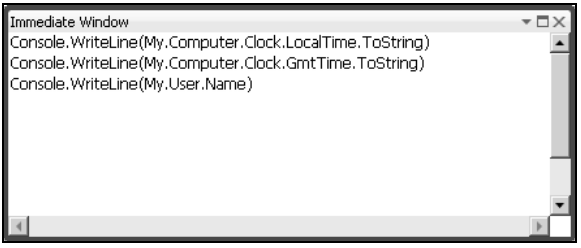

Рис. 1.12. Окно **Immediate Window**

Для выполнения команды или оператора необходимо набрать строку команды и нажать клавишу <Enter>.

В окне **Immediate Window** (Окно непосредственного выполнения) так же, как и в редакторе кода, применяются всплывающие подсказки.

Окно *Watch*

Для более полного контроля работы приложения используется окно **Watch** (Наблюдение) — рис. 1.13. Это окно вызывается командой **Watch** (Наблюдение) из подменю **Windows** (Окна) меню **Debug** (Отладка) и предназначено для определения значений выражений.

В окне **Watch** (Наблюдение) можно выполнять действия, аналогичные реализуемым в окне **Locals** (Локальные переменные).

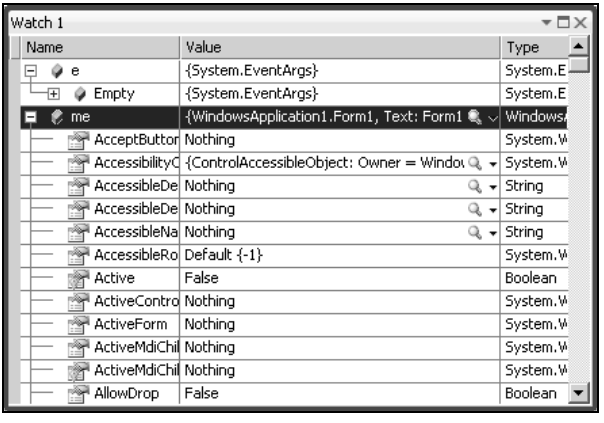

Рис. 1.13. Окно **Watch**

Справочная система

При разработке приложений неоднократно возникает необходимость просмотра возможностей средств программирования, отдельных команд и функций. В Visual Basic, кроме традиционной справочной системы, вы можете найти интересующую вас информацию в многочисленных примерах, а также на Web-страницах. Все эти средства доступны из меню **Help** (Справка).

Окно справочной системы

Окно справки отображается в браузере и имеет показанный на рис. 1.14 вид. Это окно разделено на две области. В левой области окна справочной системы расположен поиск требуемой информации и ссылки по теме. Правая область окна содержит информацию выбранного раздела.

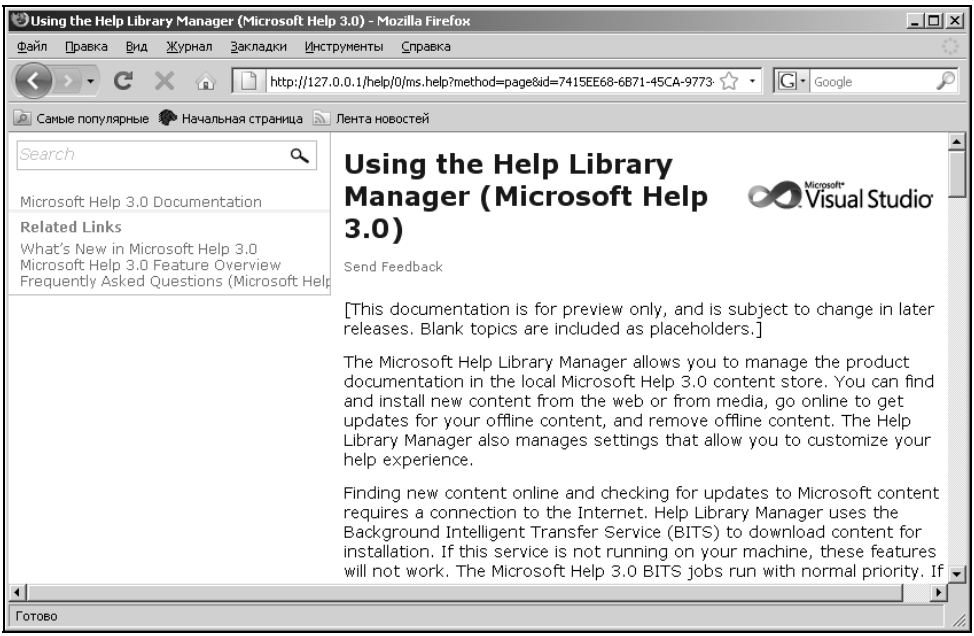

Рис. 1.14. Справочная система

Настройка справочной системы

Окно **Help Library Manager** (Управление библиотекой помощи) предназначено для настройки справочной системы (рис. 1.15). Вызвать его можно командой **Manage Help Settings** (Управление настройками справки) из меню **Help** (Справка).

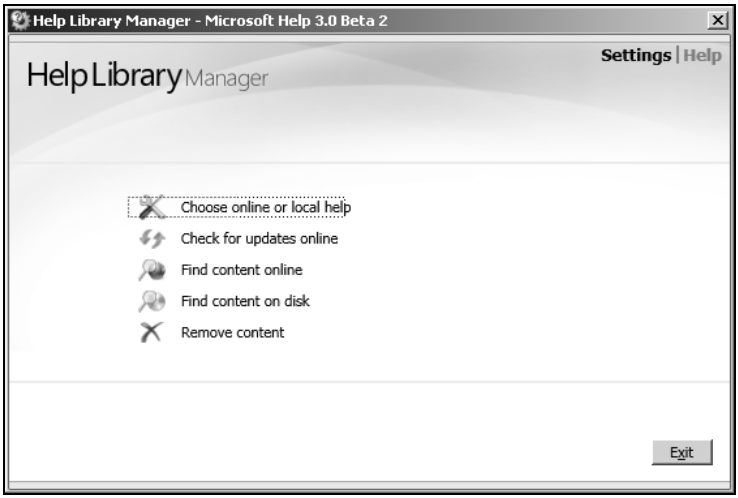

Рис. 1.15. Окно **Help Library Manager**

С помощью команды **Choose online or local help** (Выберите Интернет или локальную справку) можно воспользоваться справкой либо из Интернета, либо локальной (рис. 1.16). При использовании локальной справки указывается ее местоположение на компьютере.

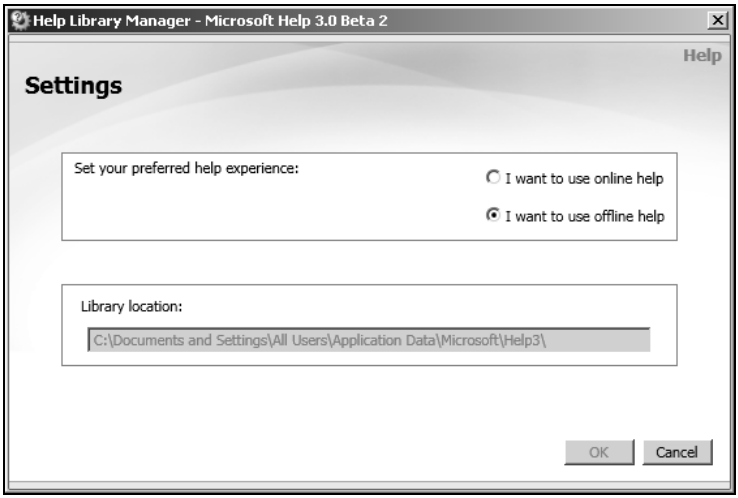

Рис. 1.16. Выбор интерактивной или локальной справки

При работе со справкой возможно скачивание новой информации с Web-сайта Microsoft для использования в автономном режиме. Для этой функции требуется подключение к Интернету. Для поиска новой информации предна-

значена команда **Find Content Online** (Найти содержимое в Интернете). Она показывает доступную для скачивания справочную информацию и уже установленную на компьютере (рис. 1.17).

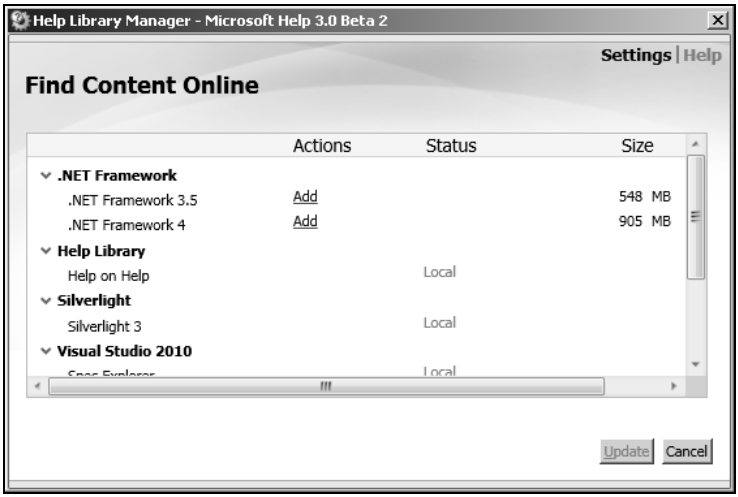

Рис. 1.17. Доступны компоненты справочной системы

Для поиска справочной информации на компьютере предназначена команда **Find Content on Disc** (Найти содержимое на диске). Она показывает доступную для установки справочную информацию и уже установленную на компьютере (рис. 1.18).

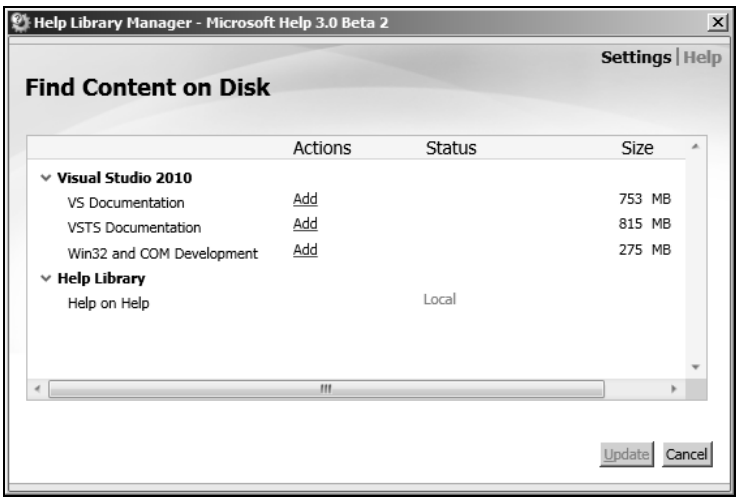

Рис. 1.18. Установленная и доступная для установки информация справочной системы

Справочная система поддерживает загрузку обновленной информации из Интернета и ее установку для просмотра в автономном режиме. Для этого используется команда **Check for updates online** (Проверка обновлений). Она показывает уже установленное содержимое, его статус и общий размер файла обновления (рис. 1.19).

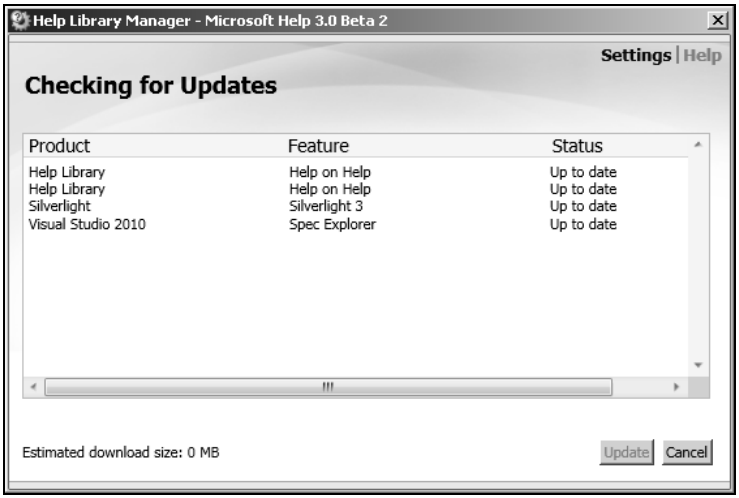

Рис. 1.19. Окно после выбора команды **Check for updates online**

Для удаления содержимого справки предназначена команда **Remove content** (Удалить содержимое). Она показывает уже установленное содержимое, его статус и размер содержимого справки (рис. 1.20).

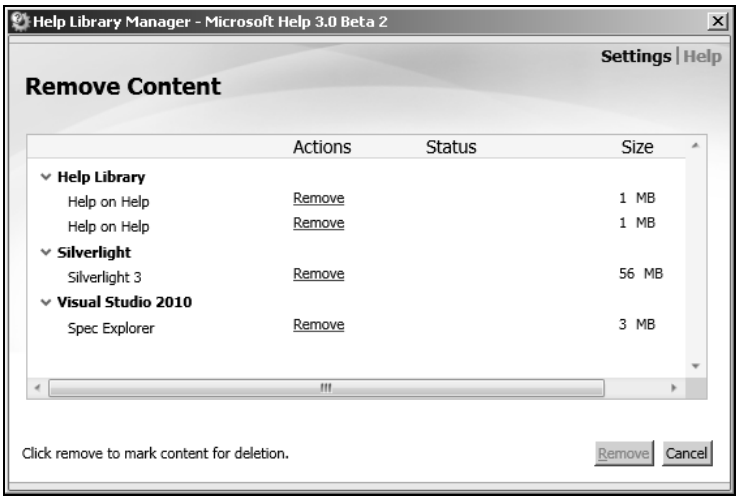

Рис. 1.20. Окно после выбора команды **Remove content**

ГЛАВА 2

Основы программирования в Visual Basic 2010

В этой главе рассматриваются основные элементы Visual Basic, используемые при создании программ.

Переменные

Переменная представляет собой зарезервированное место в оперативной памяти для временного хранения данных. Каждая переменная имеет собственное имя. После того как переменной присвоено значение, вы можете в программе вместо самого значения использовать эту переменную.

Имена переменных

Для того чтобы сделать ваши переменные более наглядными и простыми для чтения, рекомендуется давать им имена, имеющие определенное смысловое значение. Существует несколько правил задания имен переменных:

- имя переменной может содержать любые буквы, цифры и символ подчеркивания;
- первый символ в имени переменной должен быть буквой или символом подчеркивания;
- в имени переменной должны отсутствовать пробелы и знаки пунктуации;
- имя должно быть уникальным внутри области видимости;
- имя не должно являться ключевым словом, например, `Print`.

Замечание

Список ограничений достаточно велик, чтобы знать его наизусть, но вам всегда поможет проверка синтаксиса программы, при выполнении которой будет указано на использование недопустимых имен.

Например, допустимы такие имена переменных:

```
CurrentNum, Total, Date_of_birth
```

Следующие имена недопустимы:

```
2Time, $Total, Date of birth
```

Типы данных

Основные типы данных Visual Basic приведены в табл. 2.1.

Таблица 2.1. *Типы данных Visual Basic*

Тип данных	Занимает в памяти	Значение	Что хранит
`Boolean`	Различно	`True`, `False`	Логические значения
`Byte`	1 байт	От 0 до 255 (без знака)	Двоичные числа
`Char`	2 байта	Один символ	Один символ (кодировка Unicode)
`Date`	8 байтов	Дата от 1 января 0001 года до 31 декабря 9999 года и время от 0:00:00 до 23:59:59	Значения даты и времени
`Decimal`	16 байтов	Без десятичной запятой: от –79 228 162 514 264 337 593 543 950 335 до +79 228 162 514 264 337 593 543 950 335 С десятичной запятой (28 знаков после запятой): от –7,9228162514264337593543950335 до +7,9228162514264337593543950335 Наименьшее ненулевое значение (±1E–28): ±0,0000000000000000000000000001	Число с фиксированной запятой
`Double`	8 байтов	Отрицательные числа от –1,79769313486231570E308 до 4,94065645841246544E–324; положительные числа от 4,94065645841246544E–324 до 1,79769313486231570E308	Числа с плавающей запятой двойной точности
`Integer`	4 байта	От –2 147 483 648 до 2 147 483 647	Целые числа
`Long`	8 байтов	От –9 223 372 036 854 775 808 до 9 223 372 036 854 775 807	Целые числа
`Object`	Различно	В переменной типа `Object` могут храниться значения любого типа	Ссылка на объект

Таблица 2.1 *(окончание)*

Тип данных	Занимает в памяти	Значение	Что хранит
`SByte`	1 байт	От –128 до 127	Целые числа
`Short`	2 байта	От –32 768 до 32 767	Целые числа
`Single`	4 байта	Отрицательные значения от –3,4028235E38 до –1,401298E–45; положительные значения от 1,401298E–45 до 3,4028235E38	Числа с плавающей запятой
`String`	Различно	От 0 приблизительно до 2 миллиардов символов в кодировке Unicode	Текст
`UInteger`	4 байта	От 0 до 4 294 967 295 (без знака)	Целые числа
`ULong`	8 байтов	От 0 до 18 446 744 073 709 551 615 (без знака)	Целые числа
`UShort`	2 байта	От 0 до 65 535 (без знака)	Целые числа

Переменная типа данных `Boolean` может принимать только два значения: `True` и `False`. При переводе числовых данных в логические значения 0 становится `False`, а остальные значения — `True`. Когда логические значения переводятся в числовые, `False` становится 0, а `True` — 1. По умолчанию переменной типа `Boolean` присваивается значение `False`. Названия типов, указанных в табл. 2.1, являются псевдонимами для типов, определенных в пространстве имен `System`. Так, тип `Integer` является псевдонимом для типа `System.Int32`, а тип `String` — псевдонимом для типа `System.String`. Псевдонимы полностью эквивалентны типам, объявленным в пространстве `System`.

Для хранения двоичных чисел используется переменная или массив данных типа `Byte`.

Для текстовой информации предназначены переменные типов `Char` и `String`. Первый из них хранит один символ в кодировке Unicode, а второй — строку от 0 до примерно 2 млрд символов (*строкой* называют последовательность символов, заключенную в кавычки). Переменная типа `String` является ссылкой на строку. Символы в строке не могут быть изменены, может быть изменена только ссылка на нее, что следует учитывать при написании программы.

Переменные типа `Date` хранят значения даты и времени. Значение даты должно заключаться между знаками `#` и быть в формате "месяц/день/год", например `#5/31/1993#`. По умолчанию переменные типа `Date` инициализируются значением 12:00 1 января 0001 года. Для перевода значения типа `Date` в переменную строкового типа используется функция `Format`, которая имеет следующий синтаксис:

```
Function Format(ByVal Expression As Object,
                Optional ByVal Style As String = "") As String
```

где:

- *Expression* — выражение, которое необходимо привести к строковому типу;
- *Style* — используемый при переводе к строковому типу формат (например, "dddd, MMM d yyyy" и "Long Date"). Если параметр не указан, дата представляется в наиболее коротком формате, распознаваемом компьютером, а время — в формате (12- или 24-часовом), действующем на компьютере.

Следующие строки демонстрируют использование функции `Format`:

```
Format(Now(), "Long Time")
Format(Now(), "dd.MM.yy hh:mm")
```

Для хранения целых значений служат переменные типа `Short`, `Integer` и `Long` для знаковых чисел и `UShort`, `UInteger` и `ULong` для беззнаковых. Переменные типов `Short` и `UShort` занимают меньший объем памяти, но вычисление формул, содержащих данные типа `Integer`, происходит быстрее, чем формул, содержащих данные других целых типов.

Для чисел с дробной частью предназначены типы данных `Double` и `Single`, которые хранят числа с плавающей запятой, т. е. числа, представленные в виде произведения числа (так называемые "мантиссы", как правило, в пределах от 1 до 10) на 10 в определенной степени, например, 4,5E7, что означает $4{,}5 \cdot 10^7$ или 45 000 000. Числа с плавающей запятой могут иметь и отрицательный показатель степени 10, например, 4,5E–4, что означает $4{,}5 \cdot 10^{-4}$ или 0,00045. Таким образом, числа с плавающей запятой применяются для хранения как очень малых, так и очень больших величин.

Переменные, объявленные как `Decimal`, содержат числа с фиксированной десятичной запятой. В отличие от чисел с плавающей запятой, числа данного типа не имеют множителя "десять в степени". Это позволяет избежать ошибок округления, которые могут возникнуть при обработке чисел с плавающей запятой. Поэтому рекомендуется применять тип `Decimal`, когда производятся сложные вычисления, в которых недопустима подобная погрешность.

Преобразование чисел из одних типов данных в другие может быть явным и неявным. *Неявное преобразование* выполняется автоматически при присвоении определенного значения переменной. В случае *явного преобразования* используются методы класса `System.Convert`.

Тип данных `Object` может хранить различные данные и менять их тип во время выполнения программы.

При разработке программ в среде Visual Basic в зависимости от типа данных переменных можно использовать префиксы, приведенные в табл. 2.2.