Никита Культин

Основы программирования
в Microsoft Visual C# 2010

Санкт-Петербург
«БХВ-Петербург»
2011

УДК 681.3.068+800.92Visual C# 2010
ББК 32.973.26-018.1
К90

Культин Н. Б.

К90 Основы программирования в Microsoft Visual C# 2010. — СПб.: БХВ-Петербург, 2011. — 368 с.: ил.+ CD-ROM — (Самоучитель)

ISBN 978-5-9775-0589-5

Книга является пособием для начинающих по программированию в Microsoft Visual C# 2010. В ней в доступной форме изложены принципы визуального проектирования и событийного программирования, на примерах показана технология создания программ различного назначения. Приведено описание среды разработки и базовых компонентов. Рассмотрены вопросы программирования графики, разработки программ работы с базами данных Microsoft Access и Microsoft SQL Server Compact Edition. Уделено внимание технологии LINQ, отладке программ, созданию справочной системы, установке созданной программы на компьютер пользователя. В справочнике приведено описание базовых компонентов и наиболее часто используемых функций. Прилагаемый компакт-диск содержит проекты, рассмотренные в книге.

Для начинающих программистов

УДК 681.3.068+800.92Visual C# 2010
ББК 32.973.26-018.1

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 30.12.10.
Формат 70×100 1/16. Печать офсетная. Усл. печ. л. 29,67.
Тираж 1500 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 978-5-9775-0589-5

Оглавление

Предисловие

Microsoft Visual Studio — популярнейший из инструментов разработки прикладных программ различного назначения. Одним из его компонентов является Microsoft Visual C# — среда разработки .NET-приложений. В нее на основе единого интерфейса интегрированы удобный конструктор форм, специализированный редактор кода, высокоскоростной оптимизирующий компилятор, отладчик, мастер развертывания и другие полезные инструменты.

Среда Microsoft Visual C# поддерживает технологию так называемой быстрой разработки (ее также часто называют "компонентной"). В ее основе лежит технология визуального проектирования и событийного программирования, суть которой заключается в том, что среда разработки берет на себя большую часть рутины, оставляя программисту работу по созданию диалоговых окон (визуальное проектирование) и функций обработки событий (событийное программирование).

Среда Microsoft Visual C# ориентирована на разработку .NET-приложений различного типа: Windows Forms, консольных, WPF (Windows Presentation Foundation), которые могут работать в операционных системах Windows, в том Windows Vista и Windows 7.

Технология Microsoft .NET основана на идее универсального программного кода, который может быть выполнен любым компьютером, вне зависимости от используемой операционной системы. Универсальность программного кода обеспечивается за счет предварительной (выполняемой на этапе разработки) компиляции исходной программы в универсальный промежуточный код (CIL-код, Common Intermediate Language), который во время запуска (загрузки) программы транслируется в выполняемый. Преобразование промежуточного кода в выполняемый осуществляет JIT-компилятор (от Just In Time — в тот же момент, "на лету"), являющийся элементом виртуальной выполняющей системы (Virtual Execution System, VES). Выполнение .NET-приложений в операционной системе Microsoft Windows обеспечивает Common Language Runtime (CLR, общеязыковая исполняющая среда) — компонент Microsoft .NET Framework.

Усилия Microsoft по развитию и продвижению технологии .NET привели к тому, что в общем объеме вновь создаваемого программного обеспечения различного назначения увеличивается доля .NET-приложений, программ, ориентированных на платформу .NET. Это объясняется, прежде всего, возможностями, которые предос-

тавляет платформа прикладным программам, и тем, что технология .NET поддерживается новейшими операционными системами — начиная с Microsoft Windows Vista, платформа .NET является неотъемлемой частью операционной системы.

Чтобы понять, что такое .NET, какие возможности она предоставляет программисту, необходимо опробовать ее в деле. Для этого нужно изучить среду разработки, освоить технологию разработки, понять назначение и возможности компонентов, их свойства и методы.

Книга, которую вы держите в руках, посвящена практике программирования в Microsoft Visual C# 2010, разработке приложений Windows Forms (.NET). В ней, в объеме, необходимом для начинающего программиста, приведено описание среды разработки и базовых компонентов, раскрыта суть технологии визуального проектирования и событийного программирования, показан процесс создания программ различного назначения: от простейших, до программ работы с файлами, графикой и базами данных, уделено внимание технологии LINQ.

Состоит книга из двух частей.

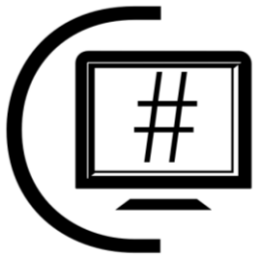
Часть I книги знакомит читателя с Microsoft Visual C++ 2010: содержит краткое описание среды разработки, демонстрирует процесс создания программы, назначение базовых компонентов.

Часть II посвящена практике. В ней рассмотрены задачи программирования графики, баз данных (Microsoft Access и Microsoft SQL Compact Edition), использования технологии LINQ для работы с массивами и XML-документами, создания справочной системы и развертывания приложений на основе технологии ClickOnce. Уделено внимание разработке консольных приложений. В главе-справочнике приведено описание базовых компонентов и наиболее часто используемых функций.

Цель этой книги — научить читателя программировать в среде Microsoft Visual C#, создавать программы различного назначения: от простых однооконных приложений до программ работы с базами данных. Следует обратить внимание на то, что хотя книга ориентирована на читателя, обладающего начальными знаниями и опытом в программировании, она вполне доступна и начинающим.

Научиться программировать можно только программируя. Поэтому, чтобы получить максимальную пользу от книги, вы должны работать с ней активно. Изучайте листинги, старайтесь понять, что и как делают программы, как они работают. Не бойтесь экспериментировать — вносите в программы изменения. Чем больше вы сделаете самостоятельно, тем большему научитесь!

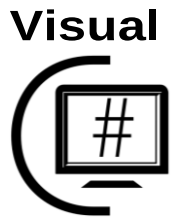
Visual



ЧАСТЬ I

Microsoft Visual C# 2010

В этой части книги приведено краткое описание среды разработки Microsoft Visual C# 2010; на примере программы "Конвертер" показан процесс разработки приложения Windows Forms; дано описание и приведены примеры использования базовых компонентов.



ГЛАВА 1

Среда разработки Microsoft Visual C# 2010

Установка

Среда разработки Microsoft Visual Studio 2010, в которую входит Microsoft Visual C#, доступна в трех вариантах: Professional, Premium и Ultimate. Каждый комплект представляет собой набор инструментов, обеспечивающих разработку высокоэффективных приложений для Win32- и .NET-платформ, Web-приложений. Различие вариантов состоит в составе компонентов — чем выше уровень (от Professional к Ultimate), тем больше возможностей и инструментов поддержки процесса разработки он предоставляет программисту. Так, например, в Ultimate включены инструменты UML-моделирования и анализа, в версии Premium есть только возможность просмотра UML-диаграмм, а в версии Professional работа с UML-диаграммами не поддерживается.

> ***Замечание***
>
> Помимо перечисленных выше коммерческих версий Microsoft Visual Studio 2010 корпорация Microsoft бесплатно предоставляет всем желающим Express-версию продукта. Отличие версии Express от коммерческих состоит в том, что она предназначена исключительно для целей обучения, *изучения* среды разработки и не дает право разработки коммерческих продуктов. Загрузить Microsoft Visual Studio 2010 Express целиком или только отдельные ее компоненты (например, Microsoft Visual C# 2010 Express) можно с узла **http://miscrosoft.com/express** (**http://www.microsoft.com/express/Downloads/#2010-Visual-CS**). Как и коммерческие версии Visual Studio, версия Express требует активации. Ключ активации предоставляется бесплатно. При этом следует учитывать, что лицензия на использование Express-версии именная, т. е. предоставляется конкретному физическому лицу.

Microsoft Visual Studio 2010 может работать в среде операционных систем Microsoft:

- Windows XP (*x*86) Service Pack 3 (кроме Starter Edition);
- Windows XP (*x*64) Service Pack 2 (кроме Starter Edition);
- Windows Vista (*x*86, *x*64) Service Pack 1 (кроме Starter Edition);
- Windows 7 (*x*86, *x*64);
- Windows Server 2003 (*x*86, *x*64) Service Pack 2;

- Windows Server 2003 R2 (*x*86, *x*64);
- Windows Server 2008 (*x*86, *x*64) Service Pack 2;
- Windows Server 2008 R2 (*x*64).

Особых требований, по современным меркам, к ресурсам компьютера Microsoft Visual Studio не предъявляет: процессор должен быть с частотой не ниже 1,6 ГГц, 1 Гбайт оперативной памяти (2 Гбайт для *x*64-процессора), порядка 7 Гбайт свободного места на жестком диске.

Установка выполняется с DVD-диска, на котором помимо Visual Studio находятся все компоненты, необходимые для установки и работы среды разработки (в том числе Microsoft .NET Framework 4).

При инсталляции с DVD процесс установки активизируется автоматически, после того как установочный диск будет помещен в дисковод (если в системе автоматический запуск программ запрещен, то установщик (файл setup.exe, который находится в корне установочного диска) надо запустить вручную.

Установщик проверяет версии Windows и Microsoft .NET Framework. Если операционная система, установленная на компьютере, не соответствует требуемой, процесс установки прерывается. В случае если версия .NET Framework ниже 4, то на компьютер устанавливаются компоненты Microsoft .NET Framework 4. После этого непосредственно начинается установка Microsoft Visual Studio.

Процесс установки Visual Studio обычный. Сначала на экране появляется окно лицензионного соглашения, затем установщик предлагает выбрать вариант установки: **Full** (Полный) или **Custom** (Выборочный). Выбрав вариант установки **Custom**, в следующем окне программист может указать компоненты Visual Studio, которые надо установить.

Первый взгляд

Microsoft Visual C# позволяет создавать .NET-приложения различного типа: Windows Forms, WPF (Windows Presentation Foundation), консольные (Console). Приложения Windows Forms — это обычные приложения .NET Windows. Приложения WPF для создания пользовательских интерфейсов используют новую, построенную на основе DirectX, графическую подсистему, обладающую расширенными возможностями (прозрачность, градиентная заливка, трансформация, отображение трехмерной графики, работа с растровой и векторной графикой, работа с аудио и видео, возможность связи интерфейсных элементов с источниками данных). Консольные приложения для взаимодействия с пользователем используют консоль — окно, обеспечивающее отображение текстовой информации, и клавиатуру.

Для того чтобы приступить к созданию программы в Microsoft Visual C# или, как принято говорить, начать работу над *проектом*, надо:

1. В меню **File** выбрать команду **New Project**.
2. В открывшемся окне **New Project** (рис. 1.1) выбрать тип приложения — **Windows Forms Application Visual C#**.
3. В поле **Name** ввести имя проекта и нажать кнопку **OK**.

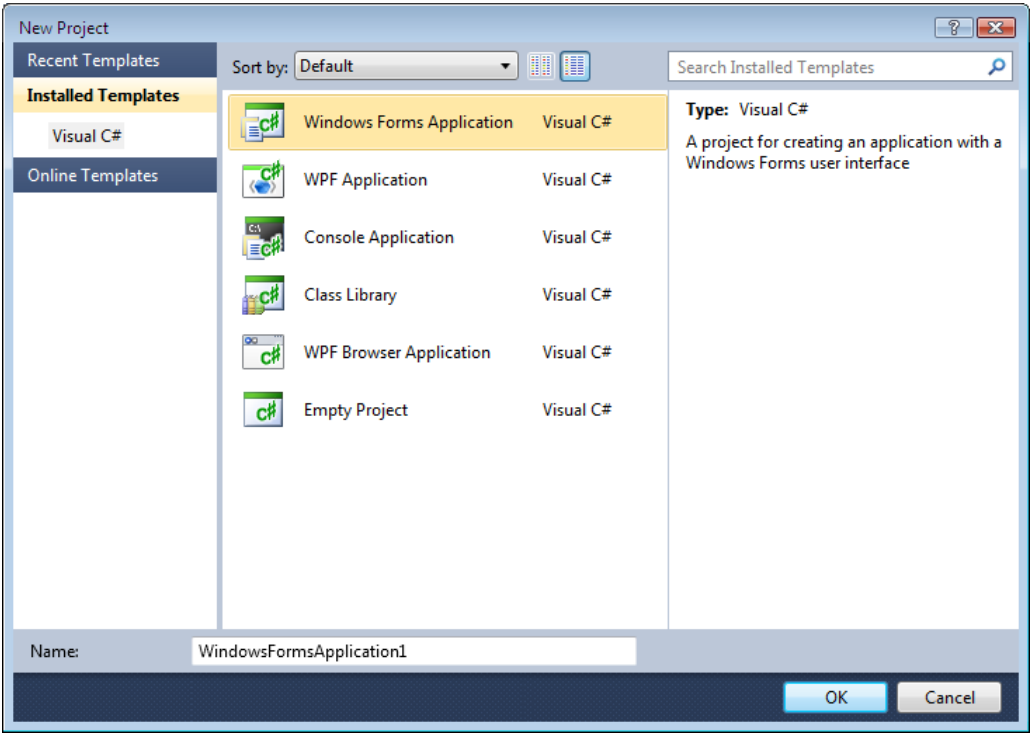


Рис. 1.1. В окне **New Project** надо указать тип приложения и задать имя проекта

В результате описанных действий будет создан *проект* — совокупность файлов, необходимых для создания компилятором выполняемого (exe) файла программы. Проект создается в папке временных проектов (по умолчанию это C:\Users***User***\AppData\Local\Temporary Projects***Project***, где: ***User*** — имя пользователя в системе; ***Project*** — имя проекта, указанное в момент его создания в окне **New Project**).

Главное окно Microsoft Visual C# в начале работы над новым проектом приведено на рис. 1.2. В его заголовке отображается имя проекта, над которым в данный момент идет работа.

В верхней части главного окна находится строка меню и область отображения панелей инструментов. По умолчанию в области панелей инструментов отображается панель **Standard** (рис. 1.3). Чтобы сделать доступными другие панели инструментов, надо в главном меню выбрать команду **View ▸ Toolbars** и в раскрывшемся списке сделать щелчок на имени нужной панели.

Центральную часть окна Visual C# занимает окно конструктора (Designer) формы (рис. 1.4). В нем находится *форма* — заготовка окна приложения (окно программы во время его разработки принято называть формой). В окне дизайнера в графической форме отображаются инструкции программы, обеспечивающие создание окна. Чтобы их увидеть (рис. 1.5), в окне **Solution Explorer** раскройте список **Form1.cs**, сделайте двойной щелчок на элементе **Form1.Designer.cs** и в открывшемся окне раскройте область **Windows Form Designer generated code** (код, сгенерированный дизайнером формы).

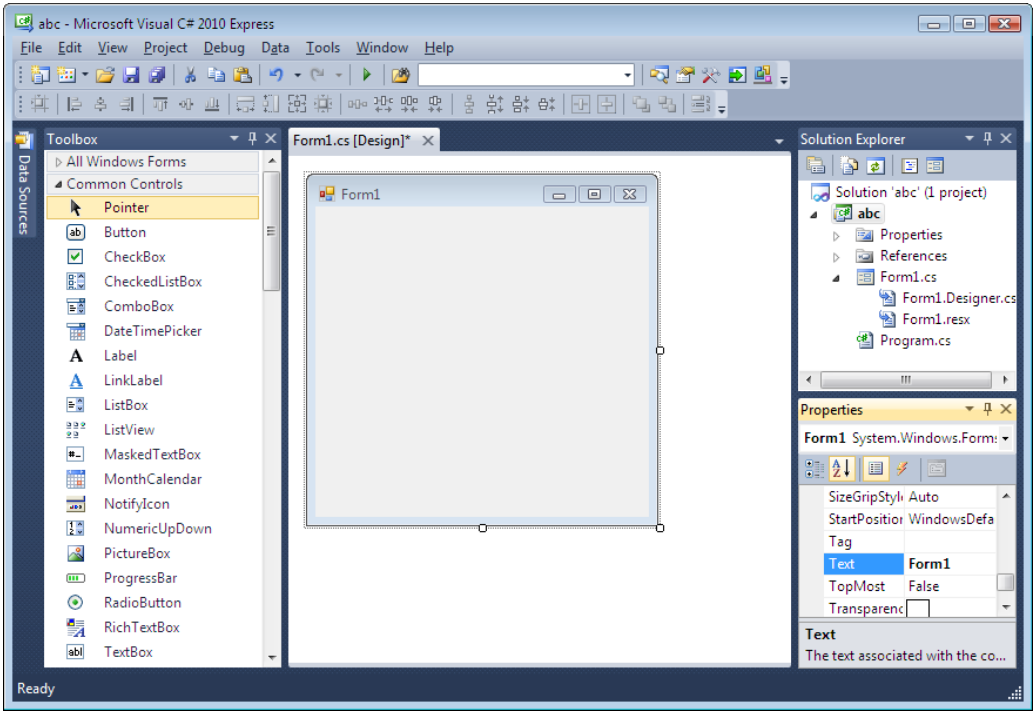


Рис. 1.2. Окно среды разработки Visual C# в начале работы над новым проектом Windows Forms

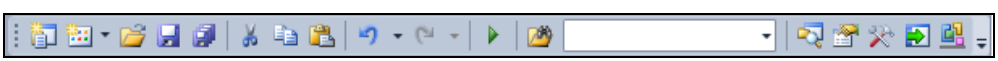


Рис. 1.3. Панель **Standard**

В левой части главного окна Visual C# отображается окно **Toolbox**. В нем (рис. 1.6) находятся *компоненты*. Компонент — это объект, реализующий некоторую функциональность. Например, в группе **Common Controls** находятся компоненты, реализующие пользовательский интерфейс (`Label` — область отображения текста; `TextBox` — поле редактирования текста; `Button` — командная кнопка), а в группе **Data** — компоненты, обеспечивающие доступ к базам данных.

В окне **Properties** (рис. 1.7) отображаются *свойства* выбранного в данный момент объекта — компонента или, если ни один из элементов (компонентов) формы не выбран, самой формы. Обратите внимание, в верхней части окна **Properties** отображаются имя выбранного объекта и его тип.

Окно **Properties** (свойства) используется для редактирования значений свойств объектов и, как следствие, изменения их вида и поведения. Например, результатом изменения значения свойства `Text` формы является изменение текста, находящегося в ее заголовке.

Свойства в окне **Properties** могут быть объединены в группы по функциональному признаку (**Categorized**) или упорядочены по алфавиту (**Alphabetical**). Чтобы изменить способ отображения, надо сделать щелчок на соответствующей кнопке, находящейся в верхней части окна **Properties** (рис. 1.8).

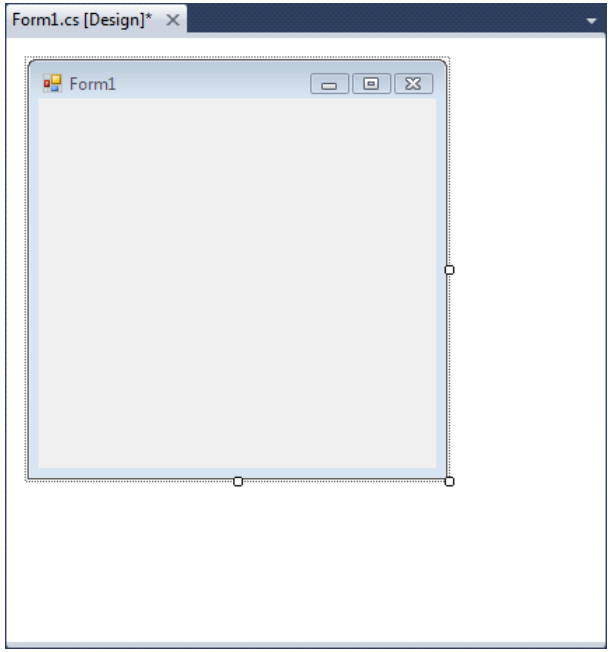


Рис. 1.4. Окно конструктора (дизайнера) формы

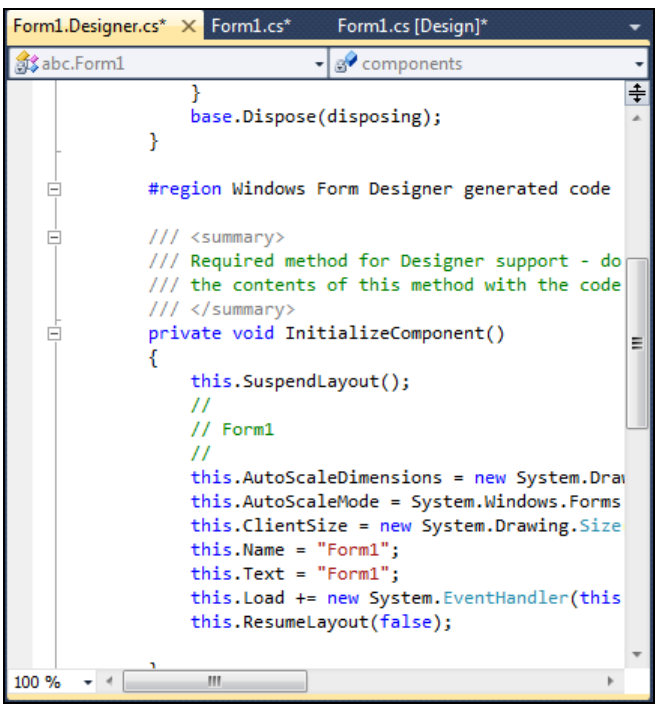


Рис. 1.5. Инструкции, обеспечивающие создание формы, находятся в секции **Windows Form Designer generated code**

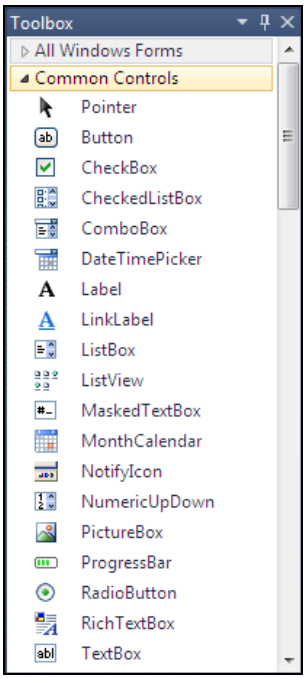


Рис. 1.6. Окно **Toolbox**

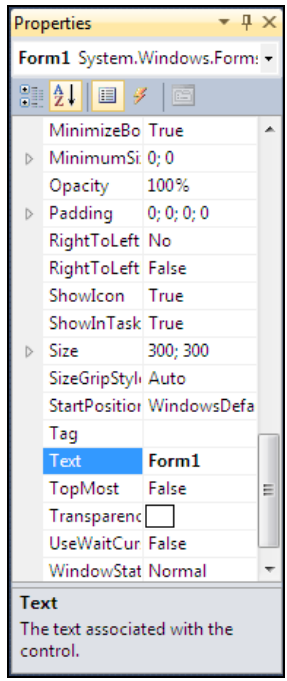


Рис. 1.7. Окно **Properties**

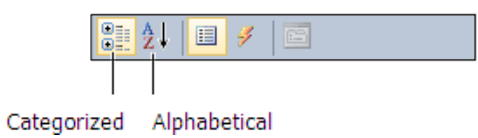


Рис. 1.8. Кнопки управления способом группировки свойств

Сделав в окне **Properties** щелчок на кнопке **Events** (рис. 1.9), можно увидеть *события* (рис. 1.10), которые способен воспринимать выбранный объект (компонент или форма). Событие — это то, что происходит во время работы программы. Например, командная кнопка может реагировать на щелчок кнопкой мыши — событие `Click`.

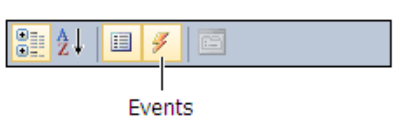


Рис. 1.9. Чтобы увидеть события, которые способен воспринимать объект, нажмите кнопку **Events**

В окне **Solution Explorer** (рис. 1.11) отображается структура проекта (solution — решение, так часто называют приложение, обеспечивающее *решение* некоторой задачи). В нем перечислены файлы, образующие проект (подробно структура про-

екта рассматривается в *главе 2*). Окно **Solution Explorer** удобно использовать для быстрого доступа к нужному элементу проекта.

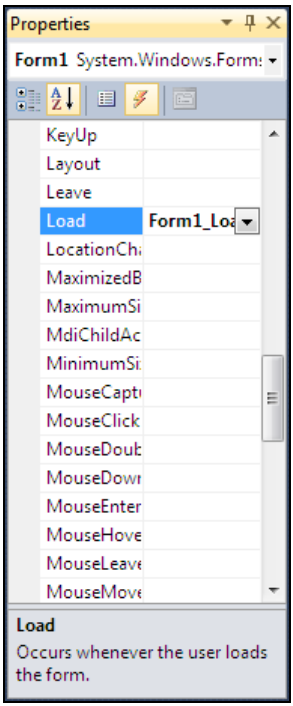


Рис. 1.10. События, которые может воспринимать объект (в данном случае — форма)

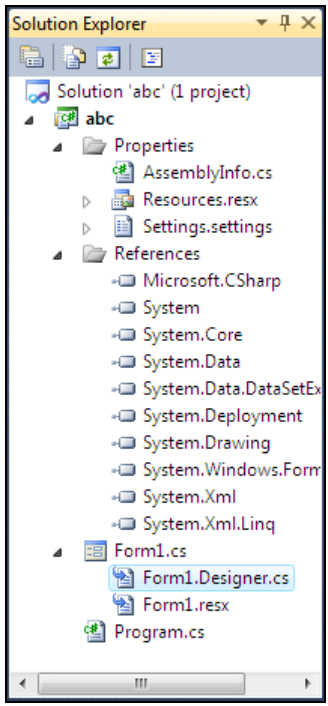


Рис. 1.11. В окне **Solution Explorer** отображается структура проекта

Справочная информация

Справочная информация Visual C# по умолчанию на компьютер программиста не устанавливается и, следовательно, недоступна. Чтобы получить доступ к справочной информации, надо выполнить настройку справочной системы.

Visual Studio позволяет программисту выбрать источник справочной информации — локальная справочная информация (находящаяся на компьютере разработчика) или online (предоставляемая Microsoft через Интернет).

Чтобы использовать локальную справочную информацию, надо:

1. В меню **Help** выбрать команду **Manage Help Settings**.
2. В окне **Help Library Manager** щелкнуть по ссылке **Choose online or local help** и в открывшемся окне выбрать **I want to use local help** (Использовать локальную справочную информацию).
3. В окне **Help Library Manager** щелкнуть по ссылке **Install Content from Online**, в появившемся окне выбрать (сделать щелчок на ссылке **Add**) раздел справочной информации, который надо установить, и нажать кнопку **Update** (рис. 1.12).

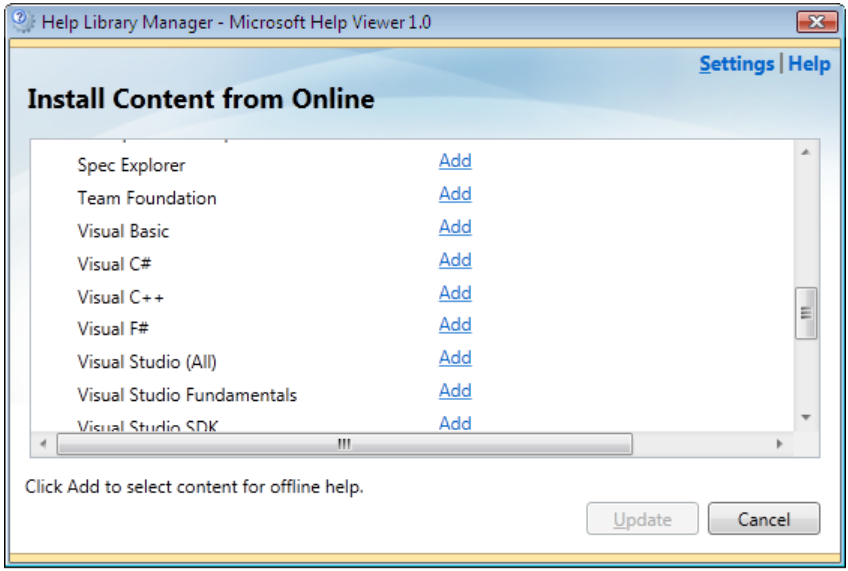


Рис. 1.12. Выбор раздела справочной информации, который надо установить на компьютер

ГЛАВА 2

Первый проект

Процесс разработки программы в Microsoft Visual C# рассмотрим на примере — создадим *приложение* (программы, предназначенные для решения прикладных задач, принято называть приложениями), позволяющее посчитать доход по вкладу (рис. 2.1).

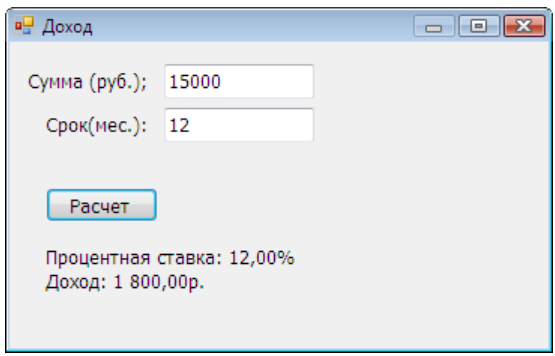


Рис. 2.1. Окно программы "Доход"

Начало работы над проектом

Чтобы начать работу над новым проектом, надо:

1. В меню **File** выбрать команду **New Project**.
2. В открывшемся окне **New Project** выбрать тип приложения — **Windows Forms Application - Visual C#**.
3. В поле **Name** ввести имя проекта — **profit** и нажать кнопку **OK** (рис. 2.2).

В результате описанных действий в папке временных проектов (по умолчанию это C:\Users***User***\AppData\Local\Temporary Projects) будет создана папка profit, а в ней — проект profit.

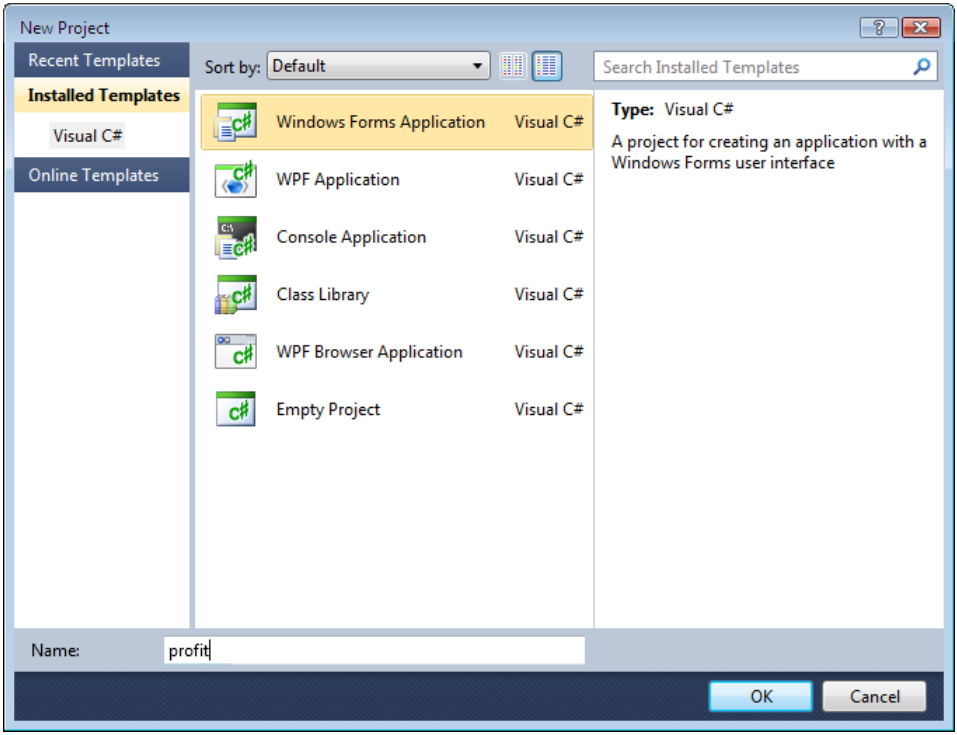


Рис. 2.2. Начало работы над новой программой

Форма

Работа над приложением начинается с создания стартовой *формы* — главного окна программы. Форма создается путем добавления на заготовку формы необходимых компонентов и изменения значений свойств самой формы.

Сначала нужно установить требуемые значения свойств формы, затем — поместить на форму необходимые *компоненты* (поля ввода информации, командные кнопки, поля отображения текста и др.) и выполнить.

Настройка формы (а также компонентов) осуществляется путем изменения значений *свойств*. Свойства *объекта* (формы, компонента) определяют его вид и поведение. Например, свойство `Text` определяет текст заголовка окна, а свойство `StartPosition` — положение окна в момент появления его на экране.

Основные свойства формы (объекта `Form`) приведены в табл. 2.1.

Таблица 2.1. *Свойства формы (объекта* `Form`*)*

Свойство	Описание
`Name`	Имя формы
`Text`	Текст в заголовке

Таблица 2.1 *(окончание)*

Свойство	Описание
Size	Размер формы. Уточняющее свойство Width определяет ширину, свойство Heigh — высоту
StartPosition	Положение формы в момент первого появления на экране. Форма может находиться в центре экрана (CenterScreen), в центре родительского окна (CenterParent). Если значение свойства равно Manual, то положение формы определяется значением свойства Location
Location	Положение формы на экране. Расстояние от верхней границы формы до верхней границы экрана задает уточняющее свойство Y, расстояние от левой границы формы до левой границы экрана — уточняющее свойство X
FormBorderStyle	Тип формы (границы). Форма может представлять собой обычное окно (Sizable), окно фиксированного размера (FixedSingle, Fixed3D), диалог (FixedDialog) или окно без кнопок **Свернуть** и **Развернуть** (SizeableToolWindow, FixedToolWindow). Если свойству присвоить значение None, у окна не будет заголовка и границы
ControlBox	Управляет отображением системного меню и кнопок управления окном. Если значение свойства равно False, то в заголовке окна кнопка системного меню, а также кнопки **Свернуть**, **Развернуть**, **Закрыть** не отображаются
MaximazeBox	Кнопка **Развернуть**. Если значение свойства равно False, то находящаяся в заголовке окна кнопка **Развернуть** недоступна
MinimazeBox	Кнопка **Свернуть**. Если значение свойства равно False, то находящаяся в заголовке окна кнопка **Свернуть** недоступна
Icon	Значок в заголовке окна
Font	Шрифт, используемый по умолчанию компонентами, находящимися на поверхности формы. Изменение значения свойства приводит к автоматическому изменению значения свойства Font всех компонентов формы (при условии, что значение свойства компонента не было задано явно)
ForeColor	Цвет, наследуемый компонентами формы и используемый ими для отображения текста. Изменение значения свойства приводит к автоматическому изменению соответствующего свойства всех компонентов формы (при условии, что значение свойства Font компонента не было задано явно)
BackColor	Цвет фона. Можно задать явно (выбрать на вкладке **Custom** или **Web**) или указать элемент цветовой схемы (выбрать на вкладке **System**)
Opacity	Степень прозрачности формы. Форма может быть непрозрачной (100%) или прозрачной. Если значение свойства меньше 100%, то сквозь форму видна поверхность, на которой она отображается

Для изменения значений свойств объектов используется окно **Properties**. В левой колонке окна перечислены свойства объекта, *выбранного* в данный момент, в правой — указаны значения свойств. Имя выбранного объекта отображается в верхней части окна **Properties**.

Чтобы в заголовке окна отображалось название программы, надо изменить значение свойства `Text`. Для этого следует щелкнуть левой кнопкой мыши в поле значение свойства `Text` (в поле появится курсор), ввести в поле редактирования текст **Доход** и нажать клавишу <Enter> (рис. 2.3).

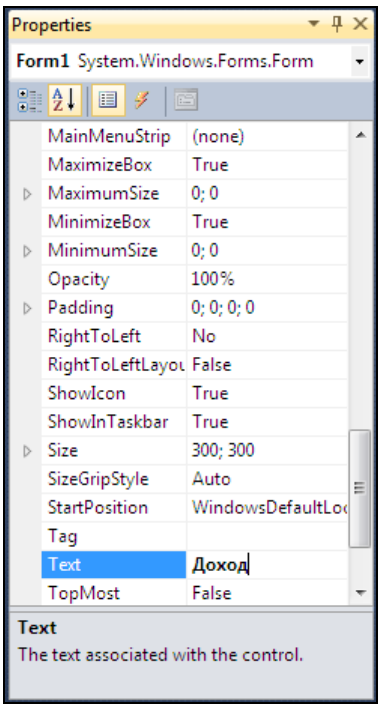


Рис. 2.3. Изменение значения свойства `Text` путем ввода нового значения

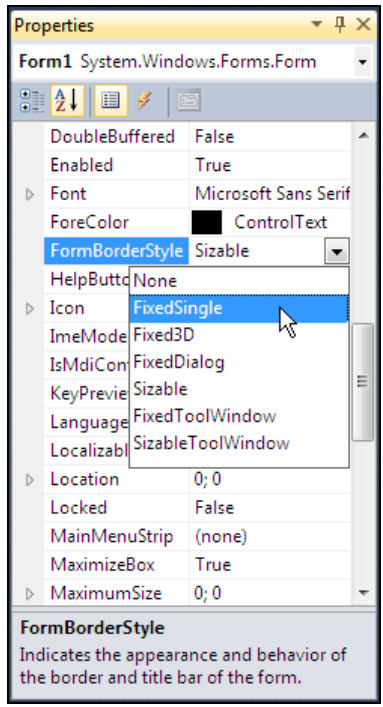


Рис. 2.4. Установка значения свойства путем выбора из списка

При выборе некоторых свойств, например `FormBorderStyle`, справа от текущего значения свойства появляется значок раскрывающегося списка. Очевидно, что значение таких свойств можно задать путем выбора из списка (рис. 2.4).

Некоторые свойства являются сложными. Они представляют собой совокупность других (уточняющих) свойств. Например, свойство `Size`, определяющее размер формы, представляет собой совокупность свойств `Width` и `Height`. Перед именами сложных свойств стоит значок ▸, в результате щелчка на котором раскрывается список уточняющих свойств (рис. 2.5). Значение уточняющего свойства можно задать (изменить) обычным образом — ввести нужное значение в поле редактирования.

Размер формы можно изменить и с помощью мыши, точно так же, как и любого окна, т. е. путем перемещения границы. По окончании перемещения границы значения свойств `Width` и `Height` будут соответствовать установленному размеру формы.

В результате выбора некоторых свойств, например `Font`, в поле значения свойства отображается кнопка, на которой изображены три точки. Это значит, что задать

значение свойства можно в дополнительном диалоговом окне, которое появится в результате щелчка на этой кнопке. Например, значение свойства `Font` можно задать путем ввода значений уточняющих свойств (`Name`, `Size`, `Style` и др.), а можно воспользоваться стандартным диалоговым окном **Шрифт**, которое появится в результате щелчка на кнопке с тремя точками (рис. 2.6).

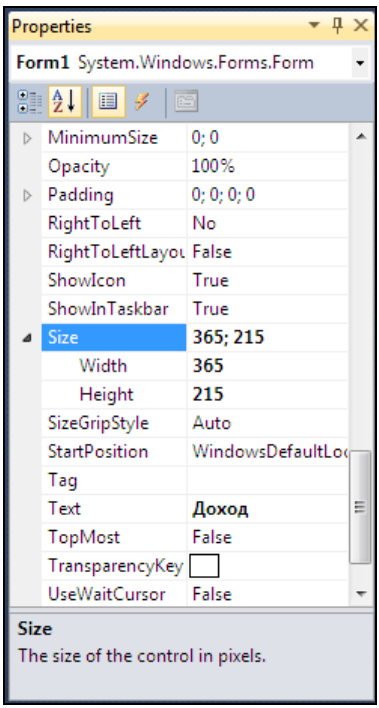


Рис. 2.5. Изменение значения уточняющего свойства

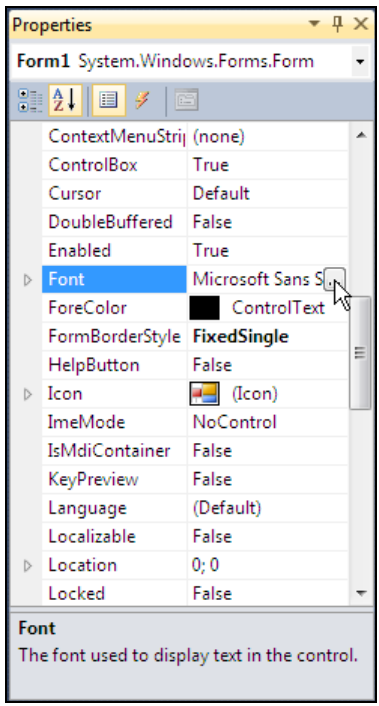


Рис. 2.6. Чтобы задать свойства шрифта, щелкните на кнопке с тремя точками

В табл. 2.2 приведены значения свойств формы программы "Доход". Значения остальных свойств формы оставлены без изменения и поэтому в таблице не представлены. Обратите внимание, в именах некоторых свойств есть точка. Это значит, что это значение уточняющего свойства.

Таблица 2.2. *Значения свойств стартовой формы*

Свойство	Значение	Комментарий
`Text`	`Доход`	
`Size.Width`	`365`	
`Size.Height`	`215`	
`FormBorderStyle`	`FixedSingle`	Тонкая граница формы. Во время работы программы пользователь не сможет изменить размер окна путем захвата и перемещения его границы

Таблица 2.2 *(окончание)*

Свойство	Значение	Комментарий
`StartPosition`	`CenterScreen`	Окно программы появится в центре экрана
`MaximizeBox`	`False`	В заголовке окна не отображать кнопку **Развернуть**
`Font.Name`	`Tahoma`	
`Font.Size`	`10`	

После того как будут установлены значения свойств формы, она должна выглядеть так, как показано на рис. 2.7. Теперь на форму надо добавить *компоненты*.

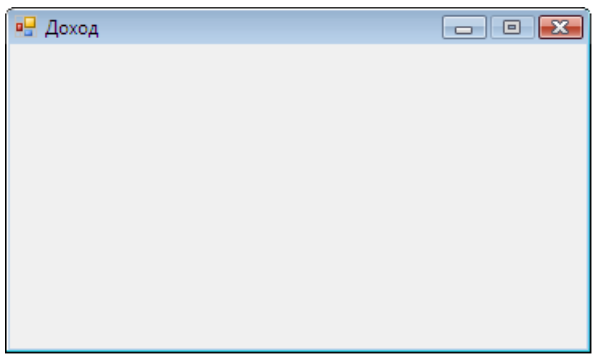


Рис. 2.7. Форма после изменения значений ее свойств

Компоненты

Поля ввода/редактирования, поля отображения текста, командные кнопки, списки, переключатели и другие элементы, обеспечивающие взаимодействие пользователя с программой, называют *компонентами пользовательского интерфейса*. Они находятся в окне **Tollbox** на вкладке **Common Controls**.

Программа "Доход" должна получить от пользователя исходные данные — сумму и срок вклада. Ввод данных с клавиатуры обеспечивает компонент `TextBox`. Таким образом, на форму разрабатываемого приложения нужно поместить два компонента `TextBox`.

Чтобы на форму добавить компонент `TextBox`, надо:

1. В палитре компонентов (окно **Toolbox**) раскрыть вкладку **Common Controls**.
2. Сделать щелчок на значке компонента `TextBox` (рис. 2.8).
3. Установить указатель мыши в ту точку формы, в которой должен быть левый верхний угол компонента, и сделать щелчок левой кнопкой мыши.

В результате на форме появляется поле ввода/редактирования — компонент `TextBox` (рис. 2.9).

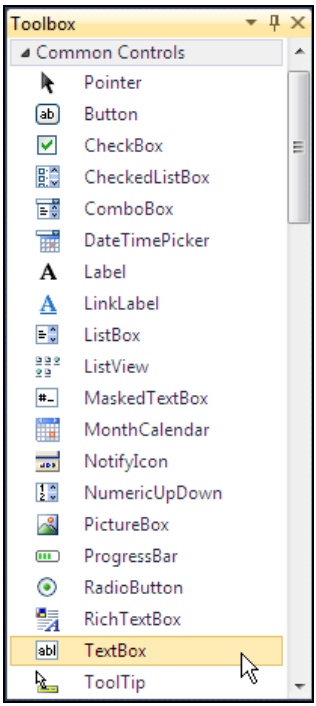


Рис. 2.8. Выбор компонента в палитре (компонент `TextBox` — поле редактирования)

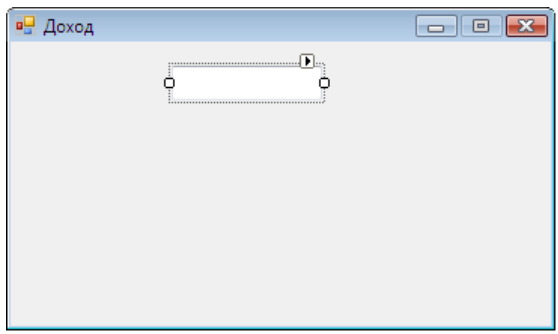


Рис. 2.9. Результат добавления на форму компонента `TextBox`

Каждому добавленному компоненту среда разработки присваивает имя, которое состоит из названия компонента и его порядкового номера. Например, первый добавленный на форму компонент `TextBox` получает имя `textBox1`, второй — `textBox2`. Программист путем изменения значения свойства `Name` может поменять имя компонента. Однако в простых программах имена компонентов, как правило, не меняют.

Основные свойства компонента `TextBox` приведены в табл. 2.3.

***Таблица 2.3.** Свойства компонента `TextBox`*

Свойство	Описание
`Name`	Имя компонента. Используется для доступа к компоненту и его свойствам
`Text`	Текст, который находится в поле редактирования
`Location`	Положение компонента на поверхности формы
`Size`	Размер компонента
`Font`	Шрифт, используемый для отображения текста в поле компонента
`ForeColor`	Цвет текста, находящегося в поле компонента
`BackColor`	Цвет фона поля компонента
`BorderStyle`	Вид рамки (границы) компонента. Граница компонента может быть обычной (`Fixed3D`), тонкой (`FixedSingle`) или отсутствовать (`None`)

Таблица 2.3 *(окончание)*

Свойство	Описание
`TextAlign`	Способ выравнивания текста в поле компонента. Текст в поле компонента может быть прижат к левой границе компонента (`Left`), правой (`Right`) или находиться по центру (`Center`)
`MaxLength`	Максимальное количество символов, которое можно ввести в поле компонента
`Multiline`	Разрешает (`True`) или запрещает (`False`) ввод нескольких строк текста
`ReadOnly`	Разрешает (`True`) или запрещает (`False`) редактирование отображаемого текста
`Lines`	Массив строк, элементы которого содержат текст, находящийся в поле редактирования, если компонент находится в режиме `MultiLine`. Доступ к строке осуществляется по номеру. Строки нумеруются с нуля
`ScrollBars`	Задает отображаемые полосы прокрутки: `Horizontal` — горизонтальная; `Vertical` — вертикальная; `Both` — горизонтальная и вертикальная; `None` — не отображать

На рис. 2.10 приведен вид формы программы "Доход" после добавления двух полей ввода/редактирования. Один из компонентов *выбран* — выделен рамкой. Свойства именно этого (выбранного) компонента отображаются в окне **Properties**. Чтобы увидеть и, если надо, изменить свойства другого компонента, нужно этот компонент выбрать — щелкнуть левой кнопкой мыши на изображении компонента в форме или выбрать его имя в раскрывающемся списке, который находится в верхней части окна **Properties** (рис. 2.11).

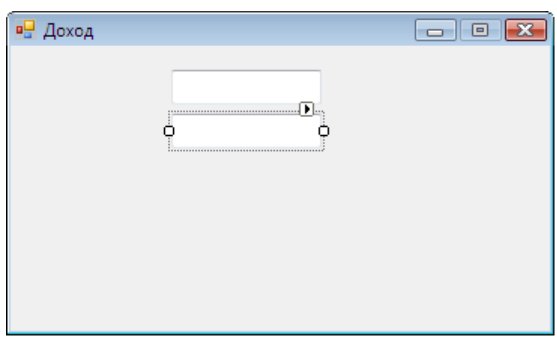


Рис. 2.10. Форма с двумя компонентами `TextBox`

Значения свойств компонента, определяющих размер и положение компонента на поверхности формы, можно изменить с помощью мыши.

Чтобы изменить положение компонента, необходимо установить курсор мыши на его изображение, нажать левую кнопку мыши и, удерживая ее нажатой, переместить компонент в нужную точку формы (рис. 2.12).

Для того чтобы изменить размер компонента, необходимо сделать щелчок на его изображении (в результате чего компонент будет выделен), установить указатель

мыши на один из маркеров, помечающих границу компонента, нажать левую кнопку мыши и, удерживая ее нажатой, изменить положение границы компонента (рис. 2.13).

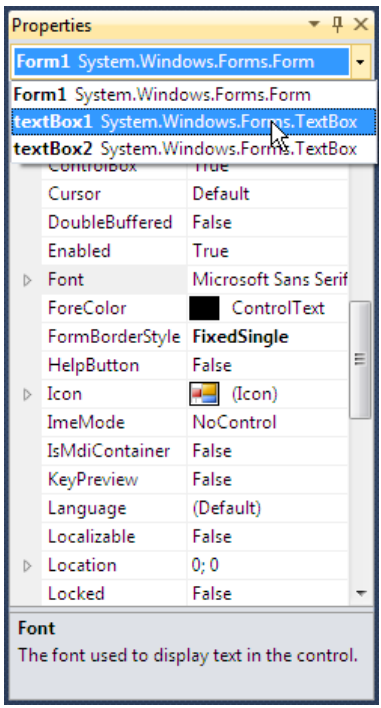


Рис. 2.11. Выбор компонента в окне **Properties**

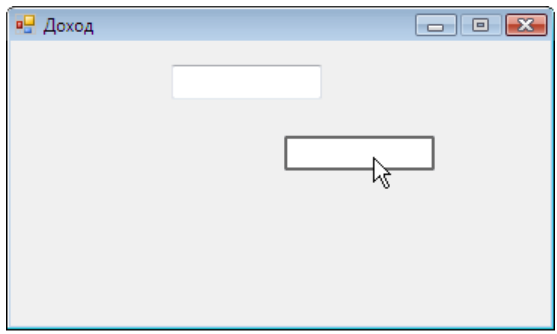


Рис. 2.12. Изменение положения компонента

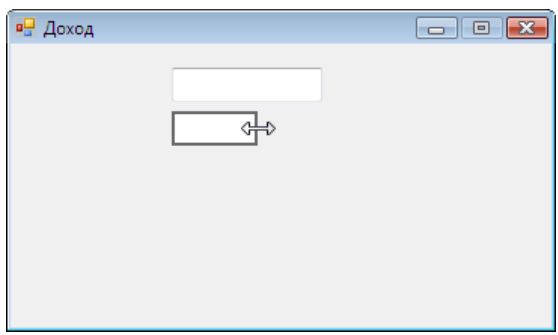


Рис. 2.13. Изменение размера компонента

В табл. 2.4 приведены значения свойств компонентов `textBox1` и `textBox2` (прочерк показывает, что значением свойства `Text` является пустая строка). Значения остальных свойств компонентов оставлены без изменения и поэтому в таблице не показаны. Компонент `textBox1` предназначен для ввода суммы вклада, `textBox2` — срока. Так как значения свойства `Font` компонентов `TextBox` не были изменены, то во время работы программы текст в полях редактирования будет отображаться шрифтом, заданным для формы. Компоненты `TextBox`, как и другие компоненты, находящиеся на форме, наследуют значение свойства `Font` формы (если значение свойства `Font` компонента не было задано явно). Поэтому если изменить значение свойства `Font` формы, автоматически изменятся значения свойств `Font` компонентов, находящихся на форме. Если требуется, чтобы текст в поле компонента отображался другим шрифтом, нужно явно задать значение свойства `Font` этого компонента.

Форма программы "Доход" после настройки компонентов `TextBox` приведена на рис. 2.14.

Таблица 2.4. *Значения свойств компонентов* `TextBox`

Компонент	Свойство	Значение
textBox1	Location.X	107
	Location.Y	16
	Size.Width	100
	Size.Height	23
	Text	-
	TabIndex	0
textBox2	Location.X	107
	Location.Y	45
	Size.Width	57
	Size.Height	23
	Text	-
	TabIndex	1

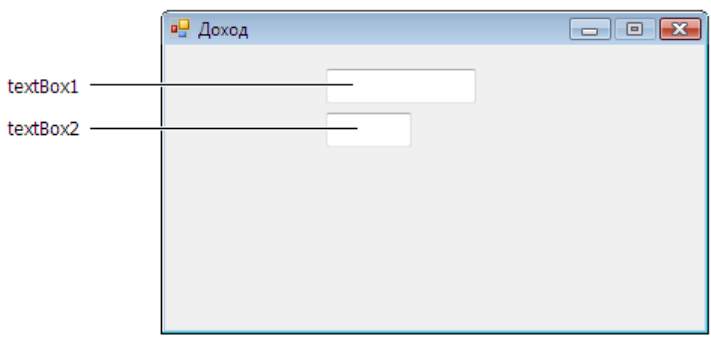


Рис. 2.14. Форма после настройки компонентов `TextBox`

Отображение текста на поверхности формы (подсказок, результата расчета) обеспечивает компонент `Label`. В окне программы "Доход" текст отображается слева от полей ввода/редактирования (информация о назначении полей). Результат расчета также отображается в окне программы. Поэтому в форму надо добавить три компонента `Label` (рис. 2.15).

Добавляется компонент `Label` на форму точно так же, как и поле редактирования (компонент `TextBox`).

Основные свойства компонента `Label` приведены в табл. 2.5.

На форму разрабатываемого приложения надо добавить три компонента `Label`. В полях `label1` и `label2` отображается информация о назначении полей ввода, поле `label3` используется для вывода результата расчета.

Значения свойств компонентов `Label` приведены в табл. 2.6.