ВЯЧЕСЛАВ ПОНАМАРЕВ

VISUAL BASIC .NET

ЭКСПРЕСС-КУРС

- Общее представление о платформе .NET
- Настройка среды разработки
- Создание приложений в VB .NET
- Разработка Windows- и Web-приложений

Вячеслав Понамарев

VISUAL BASIC .NET

ЭКСПРЕСС-КУРС

Санкт-Петербург
«БХВ-Петербург»
2003

УДК 681.3.068+800.92Visual Basic
ББК 32.973.26-018.1
П56

Понамарев В. А.

П56 Visual Basic .NET. Экспресс-курс. — СПб.: БХВ-Петербург, 2003. — 304 с.: ил.

ISBN 5-94157-340-5

В книге излагаются основные сведения об объектно-ориентированном программировании с использованием новейшей технологии .Net, успешно развиваемой ведущим разработчиком ПО Microsoft. В простой и доступной форме представлены основы языка программирования Visual Basic .Net, знакомящие читателя с синтаксисом, конструкциями языка и типами данных. Многочисленные примеры программных кодов позволяют достаточно легко перейти от простейших понятий к более серьезным, таким как классы, методы, события и их обработка. Достаточно подробно описана интегрированная среда разработки приложений Visual Studio .Net, широко используемая в последнее время для автоматизации визуального программирования. Отдельно рассмотрены возможности Visual Basic .Net при работе с графикой, дано общее представление о взаимодействии с базами данных и начальные сведения о создании Web-приложений.

Для широкого круга пользователей

УДК 681.3.068+800.92Visual Basic
ББК 32.973.26-018.1

Группа подготовки издания:

Главный редактор	*Екатерина Кондукова*
Зам. главного редактора	*Анатолий Адаменко*
Зав. редакцией	*Григорий Добин*
Редактор	*Наталья Сержантова*
Компьютерная верстка	*Ольги Сергиенко*
Корректор	*Зинаида Дмитриева*
Дизайн обложки	*Игоря Цырульникова*
Зав. производством	*Николай Тверских*

Лицензия ИД № 02429 от 24.07.00. Подписано в печать 25.07.03.
Формат 70×100 1/16. Печать офсетная. Усл. печ. л. 24,5.
Тираж 3000 экз. Заказ №
"БХВ-Петербург", 198005, Санкт-Петербург, Измайловский пр., 29.

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

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

ISBN 5-94157-340-5

Содержание

Введение

Среда разработки Visual Basic .Net сильно отличается от ранних версий Visual Basic. Этот язык прошел долгий путь совершенствования и теперь является достаточно развитым языком программирования. В настоящее время Visual Basic .Net стал фактически на одну ступень с такими языками программирования, как Visual C, C# и др. Более того, использование единого компилятора для этих языков позволяет создавать на Visual Basic .Net весьма серьезные приложения.

Новая технология .Net постепенно внедряется в большинство языков программирования, которые разрабатываются не только корпорацией Microsoft. Поэтому в данной книге мы немного приподнимем завесу над этой технологией.

В процессе работы над книгой использовалась версия Microsoft Visual Basic .Net и среда разработки Microsoft Development Environment Version 7.0.9466. Все примеры, приведенные в книге, тестировались именно в этой версии.

На кого рассчитана эта книга

Книга адресована всем желающим познакомиться с новой технологией .Net, продвигаемой корпорацией Microsoft, а также стремящимся научиться создавать простые программы начального уровня на языке Visual Basic .Net для Windows. Книга рассчитана на начинающих программистов, которые хотели бы изучить язык и среду Visual Basic .Net. Конечно, желательно, чтобы читатель был знаком (хотя бы поверхностно) с основами программирования на любом языке (не обязательно объектно-ориентированном). Еще лучше, если читатель знаком с программированием на языке Basic (хотя бы и самых ранних версий). Хотелось бы также, чтобы читатель был знаком с пакетом программ Microsot Office. А именно, для изучения *главы 18* необходимо знание Microsoft Access, а для изучения *главы 19* — знание программ Microsoft Word и Microsoft Excel.

Эта книга не претендует на звание справочного пособия или книги для профессионалов, она лишь познакомит вас с основами, а все остальное вы можете найти в дополнительной литературе, список которой приведен в конце книги.

Я благодарен всем читателям, которые присылают свои отзывы о моих книгах. Присылайте отзывы и пожелания по этой и другим книгам, написанным мной, по адресу: **ponamarev@mail.ru**.

Структура и особенности книги

Книга состоит из шести частей, введения и списка рекомендуемой литературы.

В *части I* книги даются начальные сведения о платформе .Net и рассказывается о языке программирования Visual Basic. Представлен общий обзор языка, синтаксис, типы данных и простые конструкции языка. Здесь же рассматриваются процедуры и функции, а также классы и их свойства.

Часть II рассказывает о среде разработки Visual Basic .Net. В данной части описывается состав интегрированной среды разработки, а также приводятся сведения о событиях и их обработке. Заключительная глава этой части посвящена рассмотрению средств отладки, применяемых в Visual Basic .Net.

Часть III книги посвящена созданию пользовательского интерфейса. Читатель научится работать с формами, стандартными и дополнительными компонентами, меню, узнает об их назначении и основных свойствах, событиях и методах. Кроме того, в этой части кратко рассматриваются стандартные графические возможности Visual Basic .Net и способы взаимодействия программы с пользователем.

Часть IV посвящена работе с файлами и созданию многопоточных приложений.

В *часть V* вошла глава, посвященная созданию приложений баз данных в Visual Basic .Net. Здесь излагаются основные принципы построения баз данных, а также способы работы с базами данных с помощью технологии ADO.Net. В этой же части рассматриваются вопросы управления другими приложениями из ваших приложений. Все это вы найдете в *главе 19*, посвященной автоматизации.

Часть VI посвящена разработке Web-приложений в Visual Basic .Net. Рассматривается язык XML и различные технологии: ASP, ASP.Net, SOAP.

В конце книги содержится список литературы, которую вам рекомендуется прочитать для углубления полученных знаний.

Соглашения, используемые в книге

В книге были использованы следующие типографские соглашения:

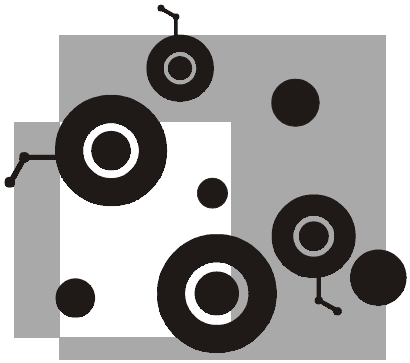
- листинги, идентификаторы, имена функций, классов, переменных, команд, т. е. любой текст листингов, который вы можете увидеть на экране монитора, выделены моноширинным шрифтом;
- *курсивом* выделены новые термины, которые впервые встречаются в тексте книги, а также важные места текста;
- **полужирным** шрифтом выделены имена элементов интерфейса: окон, пунктов меню, вкладок и т. д.;
- важные моменты, на которые стоит обратить внимание, выделены в примечания;
- названия клавиш клавиатуры заключены в треугольные скобки, например <F1>, для иллюстрации одновременного нажатия нескольких клавиш используется символ "+", например <Ctrl>+<Alt>+<O>.

Благодарности

Издательству "БХВ-Петербург" за содействие в написании данной книги. Особую благодарность выражаю Екатерине Кондуковой и Анатолию Адаменко.

Жене Татьяне за то, что она поддерживала и вдохновляла меня в процессе написания данной книги.

Своим родителям за поддержку, особенно на финальном этапе создания книги.



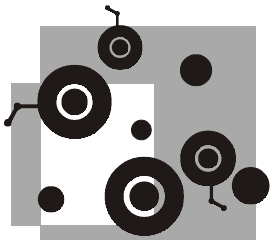
ЧАСТЬ I

Основы программирования на языке Visual Basic .Net

В этой части книги речь пойдет о синтаксических особенностях языка Visual Basic. Вы познакомитесь с историей языка Visual Basic, узнаете отличия ранних версий языка от Visual Basic .Net. Вы получите представление о том, что такое процедуры и функции, а также кратко ознакомитесь с понятием объектно-ориентированного программирования.

В начале части мы рассмотрим, что такое платформа .Net и какое место в данной платформе уделено языку Visual Basic .Net. Кроме того, вы узнаете, какие системные требования предъявляет корпорация Microsoft к работе со средой Visual Studio .Net, частью которой является Visual Basic .Net. Также вы познакомитесь с отличиями разных редакций Visual Studio .Net.

ГЛАВА 1



Платформа .Net

В этой главе вы кратко познакомитесь с новой платформой, продвигаемой компанией Microsoft — платформой .Net. Вы узнаете, что такое технология .Net. Кроме того, ознакомитесь с так называемым *промежуточным языком Intermediate Language.*

Что такое платформа .Net

Visual Basic .Net является лишь частью платформы .Net. Поэтому, прежде чем изучать Visual Basic, рассмотрим в общих чертах платформу .Net.

Платформа .Net — это совокупность компонентов и технологий, в число которых входят CLR (Common Language Runtime), ADO.Net (ActiveX Data Objects), промежуточный язык Microsoft (Intermediate Language) и др. Скажем о них несколько слов.

Среда Common Language Runtime

Среда CLR — это общая среда исполнения приложений для всех языков .Net.

Данная среда используется и для Visual Basic, и для C#, а также для многих других языков программирования. В настоящее время таких языков насчитывается более шестнадцати.

Все языки платформы .Net используют одну и ту же графическую оболочку для создания приложений, одинаковый механизм обработки исключительных ситуаций, одинаковый механизм форм и многое другое.

Преимущество данного подхода очевидно при создании больших коллективных проектов. Например, одна группа программистов пишет свою часть кода на языке Visual Basic, а другая на C#. Так как и тот и другой язык ис-

пользуют одинаковую CLR, эти разные части одного проекта можно без труда объединить. Более того, части кода, созданные на Visual Basic, можно использовать в программах, написанных на C#, и наоборот.

Еще одно преимущество CLR заключается в том, что для всех языков программирования, поддерживающих CLR, используются одинаковые средства отладки программы.

Так как программа, написанная на любом из языков .Net, и используемые ею ресурсы полностью управляются средой CLR, то такие программы называются *управляемыми*.

Примечание

Некоторые языки, например C++, могут работать без управляющей среды CLR и не будут являться управляемыми.

Промежуточный язык Microsoft

Любая программа, написанная на одном из языков платформы .Net, компилируется в программу на промежуточном языке, который получил название *Intermediate Language* (IL). Среда CLR понимает только язык IL! Таким образом, если какой-либо язык программирования можно перекомпилировать в язык IL, то этот язык будет относиться к группе .Net-языков.

В общем виде, процесс компиляции программы на .Net-языке выглядит так:

1. Создается исходный текст на базовом языке (Visual Basic .Net, Visual C++ .Net, Visual C# или другом).
2. Производится проверка синтаксиса исходного текста программы.
3. Программа перекомпилируется в промежуточный язык IL.
4. С помощью синхронного компилятора JITter программа на языке IL окончательно компилируется в машинный код для конкретного процессора.

Обращаем ваше внимание на то, что программы на языке IL не зависят от конкретного процессора. Таким образом, решается проблема переносимости кода программы с одного типа процессора на другой.

Пространство имен .Net

Платформа .Net основана на использовании классов. Так как классов огромное количество, то их имена могут повторяться. Чтобы избежать пересечения имен классов, платформа .Net использует *пространство имен*.

Пространство имен — это метод, которой применяется для создания иерархической структуры всех классов с целью избежания ситуации пересечения имен классов.

Так как пространство имен обеспечивает иерархию классов, то допускается наличие двух различных классов с одинаковым именем, но существующих в разных пространствах имен. Таким образом, пространство имен — это не что иное, как область действия класса.

Основное, базовое пространство имен платформы .Net носит название `System namespace`. Оно содержит классы для обработки исключительных ситуаций, типов данных, сборки мусора и др.

Платформа .Net содержит множество пространств имен. В табл. 1.1 перечислены некоторые из наиболее часто используемых пространств имен.

Таблица 1.1. *Часто используемые пространства имен*

Категория	Пространство имен	Описание
Компонентная модель	`System.CodeDom`	Содержит элементы и структуру исходного кода программы
	`System.ComponentModel`	Содержит описание компонентов, включая лицензионную информацию
Данные	`System.Data`	Содержит классы, обеспечивающие доступ к данным баз данных и источникам данных (классы ADO.Net)
	`System.XML`	Содержит классы, обеспечивающие поддержку языка XML
Сервисы среды	`System.Diagnostics`	Содержит классы для отладки приложений и отслеживания работы программ
	`System.Timers`	Содержит классы для обработки событий таймера
Локализация приложений	`System.Globalization`	Содержит классы, обеспечивающие поддержку создания многоязыковых приложений
Сеть	`System.Net`	Содержит классы для отправки и получения данных по компьютерной сети с использованием различных сетевых протоколов
Обычные задачи	`System.Collections`	Содержит классы, поддерживающие коллекции объектов, таких как массивы, словари и др.
	`System.IO`	Содержит классы для записи и чтения данных из потоков и файлов

Таблица 1.1 *(окончание)*

Категория	Пространство имен	Описание
(прод.)	`System.Text`	Содержит классы для работы с текстом и строками
	`System.Threading`	Содержит классы для поддержки многопоточности, включая механизмы синхронизации потоков
Графический интерфейс (GUI)	`System.Drawing`	Содержит классы для доступа к GDI и рисования двумерных картинок
	`System.Windows.Forms`	Содержит классы для создания приложений Windows, которые используют интерфейс пользователя в стиле Microsoft Windows
Безопасность	`System.Security`	Содержит классы, обеспечивающие безопасность CLR
	`System.Security.Cryptography`	Содержит классы для криптографии, включая кодирование и декодирование данных, генерирование случайных чисел и поддержку цифровых подписей
Сервисы Web	`System.Web`	Содержит классы, обеспечивающие интерфейсы взаимодействия с Web-серверами

В данной таблице перечислены далеко не все пространства имен. Если вы хотите познакомиться с остальными, обратитесь к справочным материалам по Visual Basic .Net.

Различия редакций Visual Studio .Net

Среда Visual Studio .Net поставляется конечному пользователю в четырех различных редакциях: Professional, Enterprise Developer, Enterprise Architect и Academic. В зависимости от редакции, среда Visual Studio .Net обладает теми или иными средствами. В табл. 1.2 приведены отличия разных редакций Visual Studio .Net.

Таблица 1.2. *Различия в редакциях Visual Studio .Net*

Средство/возможность	Professional	Enterprise Developer	Enterprise Architect	Academic
Visual Basic .Net	Есть	Есть	Есть	Есть
Visual C++ .Net	Есть	Есть	Есть	Есть
Visual C# .Net	Есть	Есть	Есть	Есть
Возможность создания и использования сервисов XML Web	Есть	Есть	Есть	Есть
Возможность создания Web-приложений	Есть	Есть	Есть	Есть
Возможность создания Windows-приложений	Есть	Есть	Есть	Есть
Возможность указывать дескрипторы устройств	Есть	Есть	Есть	Есть
Возможность создания таблиц и представлений с помощью SQL Server Desktop Engine	Есть	Есть	Есть	Есть
Возможность создания таблиц, представлений, процедур, триггеров, функций и т. п. для SQL Server и Oracle	Нет	Есть	Есть	Нет
Visual SourceSafe	Нет	Есть	Есть	Нет
Возможность тестирования сервисов XML Web и Web-приложений	Нет	Есть	Есть	Нет
Средства для студентов и преподавателей, включая Мастера создания приложений	Нет	Нет	Нет	Есть
Специальная документация и примеры кода для студентов и преподавателей	Нет	Нет	Нет	Есть

Системные требования для установки Visual Studio .Net

Для успешной работы со средой Visual Studio .Net корпорация Microsoft рекомендует соблюдать следующие аппаратные и программные требования (указаны минимальные требования, в скобках — рекомендуемые для комфортной работы):

- центральный процессор компьютера — Pentium II 450 МГц (Pentium III 600 МГц);
- объем оперативной памяти — в зависимости от используемой операционной системы:
  - Windows 2000 Professional — 96 Мбайт (128 Мбайт);
  - Windows 2000 Server — 192 Мбайт (256 Мбайт);
  - Windows NT 4.0 Workstation — 64 Мбайт (96 Мбайт);
  - Windows NT 4.0 Server — 160 Мбайт (192 Мбайт);
  - Windows XP Professional — 160 Мбайт (192 Мбайт);
- место на жестком диске — 500 Мбайт на системном диске + 3 Гбайт на диске для установки среды (итого 3,5 Гбайт);
- операционная система — Windows 2000, Windows XP или Windows NT 4.0;

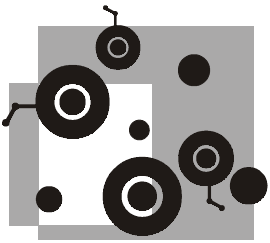
Примечание

Операционная система Windows NT 4.0 не поддерживает технологии ASP.NET, COM+, а также автоматическую сборку мусора.

- наличие привода CD-ROM или DVD-ROM;
- видеосистема с возможностью отображения 800×600 точек 256 цветами (1024×768 High Color 16-bit);
- наличие мыши.

Начиная со следующей главы этой части, вы познакомитесь с одним из представителей платформы .Net — языком Visual Basic .Net. Начнем мы с общего обзора языка Visual Basic и основных отличий языка Visual Basic .Net от предыдущих версий.

ГЛАВА 2



Общий обзор языка

В этой главе мы расскажем об истории языка Visual Basic, а также об основных трудностях, возникающих при переходе на Visual Basic .Net.

Краткая история языка Visual Basic

Изначально язык Basic был создан в качестве простого учебного языка, для освоения основ программирования вообще. Само название языка *Basic* расшифровывается как Beginners All-purpose Symbolic Instructional Code (многоцелевой код символьных инструкций для начинающих).

Язык программирования Basic появился на свет в 1964 году. Его создали два профессора из колледжа Dartmouth — Джон Кенеми и Томас Куртц для обучения студентов навыкам программирования. Язык получился настолько простым и понятным, что через некоторое время его начали применять и в других учебных заведениях. В 1975 году, с приходом первых микрокомпьютеров, эстафету развития Basic приняли Билл Гейтс и Пол Аллен, основатели Microsoft. Именно они создали новую версию Basic для первых компьютеров "Альтаир", способную работать в 4 Кбайт оперативной памяти. Со временем именно эта версия и превратилась в один из самых популярных языков программирования в мире.

На пути к популярности у Basic было множество трудностей, которые он всегда с честью преодолевал, и когда появились первые персональные компьютеры IBM PC, именно он стал стандартом в программировании, но уже в виде GW-Basic. Потом был Turbo Basic, QuickBasic, Basic PDS, но всегда при разработке новой версии языка сохранялась совместимость с прежними версиями и программа, написанная для практически первого Basic, вполне (с незначительными изменениями) могла бы работать и в последующих версиях этого языка.

Но наступили новые времена, и в начале 90-х появляется операционная система Microsoft Windows с новым *графическим интерфейсом пользователя* (GUI, Graphical User Interface). Жизнь программистов превратилась в ад. Чтобы создать простую программу, приходилось писать несколько страниц кода: создавать меню и окна, менять шрифты, очищать память, рисовать кнопки и т. д. Однако преимущества нового интерфейса были настолько неоспоримы, что уже третья версия этой операционной системы стала фактически стандартом для персонального компьютера.

Язык Visual Basic for Windows был официально представлен на выставке Windows World, которая состоялась 20 марта 1991 года. Затем язык стал совершенствоваться, выпускались новые версии под разными номерами.

Появление этого языка вызвало радость многих начинающих программистов. Он не только сохранил простоту языка Basic, но и предоставил программистам удобную *интегрированную среду разработки приложений* (IDE, Integrated Development Environment). Таким образом, с помощью IDE в Visual Basic был интегрирован набор инструментов, облегчающих и ускоряющих разработку приложений. Среда разработки Visual Basic становится в один ряд с такими средствами создания приложений, как Delphi, Visual C++ и др. Подобные среды разработки приложений получили название RAD (Rapid Application Development) — среда для быстрой разработки приложений. Действительно, создать с помощью такой среды несложное Windows-приложение — дело нескольких минут или часов.

Нужно заметить, что первые две версии языка Visual Basic были ограничены по своим возможностям. Разработать с их помощью серьезное приложение, например, для работы с базами данных, было практически невозможно.

В версии Visual Basic 3 были добавлены новые средства для работы с базами данных. В четвертой версии языка были введены базовые средства для работы с объектами, в пятой и шестой версиях появляются возможности работы с интерфейсами и расширяются возможности работы с объектами.

С помощью шестой версии языка можно было создавать приложения практически любого уровня сложности, обеспечивающие решение любых задач (базы данных, игры, мультимедиа). Простота и эффективность языка позволили сделать его встроенным языком для приложений Microsoft Office. Такой вариант языка получил название VBA — Visual Basic for Applications (язык Visual Basic для приложений).

Несмотря на все эти положительные нововведения, язык Visual Basic серьезно страдал и терял свою целостность. Сказывался тот факт, что все доработки, связанные с объектно-ориентированным подходом, производились на старом базовом фундаменте языка, в котором такая поддержка не была изначально предусмотрена. Программистам приходилось использовать особые языковые конструкции для создания новых объектов вместо конструкций, применяемых в большинстве объектно-ориентированных языков.

Таким образом, назрела необходимость серьезной переработки языка в сторону его полноценной объектной ориентации. Так был создан язык Visual Basic .Net.

Основные отличия языка Visual Basic .Net от ранних версий

Следует сразу заметить, что если вы знакомы с ранними версиями Visual Basic, то вам следует подготовиться ко многим изменениям в стиле программирования. "Обычный" Visual Basic отличается от Visual Basic .Net так же, как Visual Basic без поддержки классов от Visual Basic с поддержкой таковых. Если вы не программировали ранее в среде Visual Basic, можете смело пропустить эту главу и перейти к изучению *главы 3* данной книги.

Одним из главных отличий новой версии языка от предыдущих заключается в использовании новой исполнительной среды CLR, которую мы с вами уже рассматривали в *главе 1* этой книги.

Еще одно отличие — использование *промежуточного языка Microsoft*. О нем мы тоже упоминали ранее.

Ранние версии Visual Basic не позволяли автоматически инициализировать данные класса при создании его экземпляра. Таким образом, новый экземпляр класса создавался в неопределенном состоянии. Для обхода этого недостатка прежде программисты создавали специальную функцию (обычно с именем `Create`) и вызывали ее с целью задания значений полей экземпляра класса. В другом случае поля экземпляра класса после его создания получали значения по умолчанию. Не всегда, но довольно часто, это приводило к ошибкам в работе приложения. Причем ошибкам, которые достаточно трудно обнаружить. Новая среда Visual Basic .Net решает эту проблему с помощью так называемых *параметризованных конструкторов*. При вызове такого конструктора можно явно передавать начальные параметры новому экземпляру класса.

В предыдущих версиях языка не было полноценной реализации одного из основных принципов объектно-ориентированного программирования — *наследования*. Наследование предполагает, что программист может расширять возможности готовых классов путем создания новых объектов на базе уже существующих. Например, если возникает потребность расширить возможности стандартной кнопки, с помощью наследования можно взять готовый код класса кнопки и добавить в него новый код. Таким образом получится новый класс — расширенная кнопка, которая будет являться *потомком (производным классом)* стандартной кнопки. Новая среда разработки Visual Basic .Net позволяет полноценно применять возможности наследования в ваших приложениях.

Ранее у программистов часто возникали проблемы с *утечкой памяти* в случае *циклических ссылок*. Циклические ссылки — это когда некий объект ссылается на другой объект, а тот, в свою очередь, ссылается на первый объект. Таким образом, компилятор Visual Basic ранее не распознавал такие ссылки и не освобождал память, занимаемую подобными объектами. В среду Visual Basic .Net встроена так называемая *система сборки мусора*, которая следит за циклическими ссылками, разрывает их и освобождает занимаемую память.

В новую среду разработки встроена структурная обработка ошибок, основанная на *обработке исключительных ситуаций*. Кроме того, в Visual Basic .Net улучшена возможность создания *многопоточных приложений*. Многопоточные приложения позволяют одновременно выполнять несколько различных задач. Следует отметить, что в Visual Basic .Net не бывает однопоточных программ, поскольку одним из дополнительных потоков всегда работает система сборки мусора.

Проблемы перехода на Visual Basic .Net

Сразу необходимо отметить, что среды разработки Visual Basic 6.0 и Visual Basic .Net могут быть установлены одновременно на одном компьютере и в одной операционной системе.

Компоненты COM (Component Object Model, модель компонентных объектов Microsoft), которые были вами созданы в ранних версиях Visual Basic, могут взаимодействовать с компонентами, написанными с помощью Visual Basic .Net. Обратное взаимодействие также возможно.

Среда Visual Basic .Net не является напрямую совместимой с ранними версиями. Для преобразования прежних версий программ, написанных на ранних версиях Visual Basic, может использоваться *Мастер обновления* (Upgrade Wizard). Данный Мастер позволяет за несколько шагов преобразовать большинство проектов, написанных на ранних версиях языка, в проекты Visual Basic .Net. При таком преобразовании будет выполнена модернизация синтаксиса приложения и замена Visual Basic Forms на Windows Forms. Заметим, что проекты, написанные в среде Visual Basic .Net, не могут быть открыты в ранних версиях языка.

Многие возможности и функции ранних языков не поддерживаются новым в принципе, поэтому сложные проекты, использующие такие возможности и функции, придется переделывать самостоятельно без помощи Мастера.

Для тех читателей, которые собираются переносить свои приложения из Visual Basic в Visual Basic .Net, можно дать следующие советы:

- если вы работаете с Visual Basic версии 5.0 и ниже, то следует сначала преобразовать программы в проекты Visual Basic 6.0, а уже потом перейти к их адаптации для Visual Basic .Net;

- вполне вероятно, что некоторые из ранее созданных приложений будет выгоднее развивать и поддерживать в среде Visual Basic 6.0, чем переводить в Visual Basic .Net. Не говоря уже о том, что преобразование сложных проектов в новую среду будет совсем не мгновенным;
- будьте внимательны при копировании фрагментов кода через буфер обмена. Понятно, что в этом случае нельзя ждать автоматического преобразования кода из Visual Basic 6.0 в Visual Basic. Net — вся ответственность ложится на разработчика;
- максимально выносите логику обработки из модулей форм. Например, если у вас есть форма, которая содержит сколько-нибудь сложную логику, то имеет смысл подумать о создании специального BAS-модуля, куда можно перенести основные процедуры обработки данных. Причем речь может идти даже о двух-трех операторах из событийных процедур. Что же касается обычных процедур, то их однозначно нужно записывать в BAS-модули. Пользу такого выделения логики уже давно поняли программисты, которые работают одновременно с Visual Basic и VBA — несмотря на схожесть реализации программного интерфейса в этих двух системах, переносить программы без особых проблем удается только для модулей кода, а формы у них несовместимы. Поэтому бывает проще перерисовать форму в новой среде, адаптировать в ней минимальный код обработки и подгрузить готовый BAS-модуль. А если говорить о переносе приложений, то имеет смысл в более широком плане подумать о более четком разделении на интерфейс пользователя и бизнес-логику, а также выделении последней в виде автономных ActiveX-компонентов. Иными словами, речь может идти о выделении из единого приложения соответствующей библиотеки (или библиотек) объектов;
- ориентируйтесь на работу с ASP.Net. Visual Basic 6.0 поддерживал три типа интернет-приложений, основанных на использовании доступа через браузер: DHTML-приложения, документы ActiveX и приложения WebClass. Все эти три вида могут взаимодействовать с технологиями Visual Basic .Net. Первые два типа приложений в Visual Basic .Net не поддерживаются, и их автоматического обновления для Visual Basic .Net не предусматривается. Лучший вариант — продолжать поддерживать их в Visual Basic 6.0, хотя, возможно, имеет смысл преобразовать документы ActiveX в пользовательские элементы управления;
- используйте ADO для работы с базами данных. Главная технология доступа к данным в Visual Basic .Net — ADO.NET, представляющая собой дальнейшее развитие существующей версии ADO. Старые средства — DAO (Data Access Objects) и RDO — поддерживаются в Visual Basic .Net только на уровне кода (с некоторыми модификациями), но соответствующие элементы управления уже нельзя использовать. Таким образом, если ваши приложения применяют элементы управления DAO и RDO, то нужно либо поменять их на ADO, либо продолжать работать с ними в Visual Basic 6.0;

- не используйте свойства и методы по умолчанию. Например, для элемента управления `Label` код:

  ```
  lblMyLabel = "Мне нравится работать с Visual Basic .Net"
  ```

  в ранних версиях работал, поскольку `Caption` — это как раз такое свойство по умолчанию для метки. Хотя правильнее было бы написать (в Visual Basic 6.0):

  ```
  lblMyLabel.Caption = "Мне нравится работать с Visual Basic .Net"
  ```

  В Visual Basic .Net для метки нужно вместо `Caption` использовать свойство `Text` (оно применяется для хранения содержимого в подобных элементах управления):

  ```
  lblMyLabel.Text = "Мне нравится работать с Visual Basic .Net";
  ```

- не используйте прямые ссылки на элементы управления других форм. До сих пор все элементы управления на формах имели фиксированный статус `Public`. Соответственно, доступ к ним можно было получить из любого места проекта, например, следующим образом (к элементу `txtTitle1`, расположенному на форме `frmSomeForm`):

  ```
  sTitle = frmSomeForm.txtTitle1.Text
  ```

  В новой версии Visual Basic .Net подобным образом можно обращаться только к тем элементам управления, для которых в явном виде указан статус `Public Shared`.

  Возможно, придется изменить типы целочисленных переменных. До сих пор в Visual Basic было два типа целочисленных переменных со знаком `Integer` и `Long`. В Visual Basic .Net названия этих типов обозначают совсем другие допустимые диапазоны значений. Кроме того, добавлен новый тип. Обо всем этом более подробно вы можете узнать, прочитав *главу 3* данной книги;

- в Visual Basic .Net не поддерживается тип `Currency`, вместо него предлагается использовать тип `Decimal`;

- тип данных `Variant` больше не используется. Вместо него применяется тип `Object`, который в Visual Basic 6.0 использовался только для ссылок на объект;

- типы всех переменных должны быть описаны в явном виде (в Visual Basic 6.0 по умолчанию, при отсутствии описания `As...`, устанавливался тип `Variant`). Кроме того, в Visual Basic .Net реализовано давно необходимое (и существующее в других языках) групповое описание типов данных, например:

  ```
  Dim strFirstString, strSecondString As String.
  ```

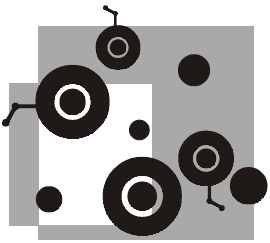
  Эта строка означает, что обе переменные — строковые и имеют тип `String`;

- заменяйте вычисляемый оператор `Goto/Gosub` на `Select`. Конструкция вычисляемого `Goto/Gosub` — типичный пережиток языка Basic 60-х годов. Многие Visual Basic-программисты ее вообще не знают и правильно делают (само использование меток внутри процедуры — это плохой стиль программирования). Аналогично, заменяйте `GoSub/Return` на `Call Sub`;
- будьте внимательны при определении массивов. Исключите использование оператора Option Base 1 (его нет в Visual Basic .Net), применяйте массивы с нулевой нижней границей индекса и пересмотрите варианты динамического определения массивов;
- откажитесь от неявного преобразования типов данных;
- забудьте о целочисленных значениях булевых переменных. Используйте значения `True` и `False`;
- используйте константы. Хороший стиль программирования подразумевает, в частности, использование внутренних глобальных констант Visual Basic вместо литералов;
- для работы с датами применяйте только тип `Date`. Во всех существовавших до сих пор версиях Visual Basic для хранения даты фактически использовались переменные типа `Double` (отдельный тип `Date` появился только в версии 4.0);
- не используйте недокументированные функции;
- не используйте оператор `LSet` для структур. В Visual Basic 6.0 этот оператор можно было применять для присвоения переменной одного пользовательского типа значения переменной другого пользовательского типа. Теперь такая возможность исключена;
- по возможности, старайтесь избегать использования строк фиксированной длины;
- точно объявляйте способ передачи параметров. Ранее способ передачи параметров — `ByRef` (по ссылке) или `ByVal` (по значению) — указывался только при работе с WinAPI (точнее, со всеми DLL-функциями). При работе с процедурами внутри среды Visual Basic исторически было принято, что по умолчанию параметры всегда передаются по ссылке. Но в Visual Basic .Net это изменилось — теперь по умолчанию все параметры передаются по значению.

В общем и целом можно сделать следующий вывод: перенос программ, написанных на Visual Basic 6.0, в среду Visual Basic .Net будет непростым, так, многие конструкции Visual Basic 6.0 не будут работать в новой версии системы. Более того, некоторые из них будут выполняться, но иначе, чем в исходной версии.

В *главе 3* мы расскажем о типах данных языка Visual Basic .Net, переменных, константах и массивах.

ГЛАВА 3



Типы данных, переменные, константы и массивы

В этой главе мы раскроем понятие типов данных. Вы узнаете, над какими типами данных допустимы операции в среде Visual Basic .Net, научитесь преобразовывать одни типы данных в другие. Кроме того, вы познакомитесь с переменными и массивами.

Понятие типов данных

Как вы знаете, все языки программирования дают возможность оперировать различными данными. Это могут быть целые и вещественные числа, строки текста, байты и многое другое. Все приведенные нами примеры данных относятся к различным типам. Они несовместимы друг с другом: ведь нельзя же, например, к предложению "Мне нравится Visual Basic .Net" прибавить число 235.23. Результат такого сложения трудно себе представить, т. к. предложение будет относиться к строковому типу данных, а число 235.23 — к типу данных вещественных чисел. Таким образом, в языках программирования существует четкое деление разнообразной информации на *типы данных*. Над различными типами данных допустимы различные операции, которые не могут выполняться над другими.

В табл. 3.1 приведены основные типы данных, с которыми может работать Visual Basic .Net. Как вы можете видеть, любые типы данных имеют свои границы, выходить за которые недопустимо.

Таблица 3.1. *Основные типы данных Visual Basic .Net*

Тип данных	Описание	Диапазон допустимых значений
`Boolean`	Булевский (логический)	`True` и `False`

Таблица 3.1 *(окончание)*

Тип данных	Описание	Диапазон допустимых значений
`Byte`	1-байтовое число	0 ... 255
`Char`	Один символ формата Unicode	0 ... 65535
`Date`	Дата и время	#01.01.0001# ... #31.12.9999#
`Decimal`	12-байтовое десятичное число	-79 228 162 514 264 337 593 543 950 335 ... 79 228 162 514 264 337 593 543 950 335
`Double`	8-байтовое вещественное число	Для отрицательных значений: –1.79 769 313 486 232 E308 ... -4.94 065 645 841 247 E-324 или для положительных значений: 4.94 065 645 841 247 E-324 ... 1.79 769 313 486 232 E308
`Integer`	4-байтовое целое число со знаком	-2 147 483 648 ... 2 147 483 647
`Long`	8-байтовое целое число со знаком	-9 223 372 036 854 775 808 ... 9 223 327 036 854 775 807
`Object`	Объект	Любое значение любого типа
`Short`	2-байтовое целое число со знаком	-32 768 ... 32 768
`Single`	4-байтовое вещественное число	Для отрицательных значений: -3.402 823 E38 ... -1.401 298 E-45 или для положительных значений: 1.401 298 E-45 ... 3.402 823 E38
`String`	Строка постоянной длины	От одного символа до двух миллионов символов приблизительно

Чтобы правильно использовать приведенные в табл. 3.1 типы данных, можно дать несколько рекомендаций:

- если вам необходимо хранить только два возможных значения, используйте тип `Boolean`;
- для хранения целых или вещественных значений используйте тот тип данных, который больше всего подходит по возможным допустимым

значениям. Всегда стремитесь, по возможности, использовать тот тип данных, который занимает меньше байт оперативной памяти;

- для хранения текстовой или смешанной информации (текст, цифры и символы) используйте тип данных `String`;
- для хранения денежных значений рекомендуется использовать тип `Decimal`;
- для хранения и работы с датой/временем используйте специальный тип данных `Date`.

Если вы забыли, какое максимальное или минимальное значение допустимо по отношению к тому или иному типу данных, вы можете воспользоваться специальными методами `MaxValue` и `MinValue` соответственно. Например:

```
Console.WriteLine(Double.MaxValue)
```

Результатом выполнения будет выдача на консоль (экран или часть экрана) максимально допустимого значения для типа данных `Double`.

Для того чтобы уточнить, к какому именно типу данных принадлежит то или иное значение, в Visual Basic .Net вы можете использовать *суффиксы*.

Суффиксы — это символы алфавита, которые следуют сразу за значением и определяют его тип.

Например, для того чтобы показать, что число 234 является вещественным, а не целым, после последней цифры числа вы можете поставить символ `F`. Запись `234F` будет говорить компилятору Visual Basic .Net о том, что число 234 относится к типу данных `Single`.

В табл. 3.2 приведены суффиксы, указывающие на различные типы данных.

Таблица 3.2. *Суффиксы типов данных*

Тип данных	Суффикс	Пример
`Char`	`C`	`"A"C`
`Decimal`	`D`	`234D`
`Double`	`R` (или `#`)	`234R`
`Integer`	`I`	`234I`
`Long`	`L`	`234L`
`Short`	`S`	`234S`
`Single`	`F`	`234F`

Использование суффиксов позволит вам избежать неприятных ошибок компиляции. Например, при выполнении команды:

```
Console.WriteLine(45621256*5741)
```

компилятор выдаст ошибку:

```
Constant expression not representable in type 'Integer'
```

Если же вы укажете с помощью суффикса тип данных, например, так:

```
Console.WriteLine(45621256L*5741)
```

ошибки не будет, а будет выдан результат вычислений:

```
261911630696
```

Скажем несколько слов о специфичном типе данных `Object`. Этот тип данных позволяет не задумываться над тем, какого типа данные в нем хранятся, поскольку `Object` позволяет хранить в себе любые данные. На этом преимущества типа данных `Object` в принципе заканчиваются. Недостатками являются большой объем памяти, расходуемой на поддержание такого типа данных, а также медленность выполнения операций над данными такого типа. Тип данных `Object` можно рекомендовать к использованию только в том случае, когда действительно неизвестно, какой тип данных вам будет необходим.

Заканчивая обозрение типов данных, приведем еще одну таблицу, в которой представлены числовые типы данных для Visual Basic .Net, Visual Basic 6.0 и .Net Framework, а также соответствия между ними.

Таблица 3.3. *Соответствия между числовыми типами данных*

Тип данных Visual Basic .Net	**Тип данных .Net Framework**	**Тип данных Visual Basic 6.0**
`Byte`	`System.Byte`	`Byte`
`Boolean`	`System.Boolean`	`Boolean`
`Decimal`	`System.Decimal`	Нет аналога
Нет аналога	Нет аналога	`Currency` (денежный)
`Double`	`System.Double`	`Double`
`Short`	`System.Int16`	`Integer`
`Integer`	`System.Int32`	`Long`
`Long`	`System.Int64`	Нет аналога
`Single`	`System.Single`	`Single`

Как видно из табл. 3.3, некоторые названия типов данных были изменены, а некоторые поменяли свое значение. Если вы ранее программировали