Часть XI

Программный потенциал VBA

В этой части...

- Глава 51. Основы Visual Basic for Application
- Глава 52. Использование редактора Visual Basic
- Глава 53. Написание больших модулей VBA
- Глава 54. Объектно-ориентированное программирование на VBA
- Глава 55. Надежный код: отладка и перехват ошибок
- Глава 56. Становимся интерактивными: специальные диалоговые окна
- Глава 57. Создание мощных приложений

Visual Basic for Application — мощный язык программирования, инструмент, позволяющий добраться до скрытых ресурсов MS Office. В этой части вы пройдете курс изучения VBA, который начинается ознакомлением с самим языком VBA (и редактором Visual Basic, поставляющимся с Office XP) и включает изучение аспектов, связанных с программированием (таких, как модули, объектно-ориентированное программирование, отладка программ и поиск ошибок, пользовательские диалоговые окна и т.д.).

Глава 51

Основы Visual Basic for Application

В этой главе...

- Знакомство с редактором Visual Basic
- Написание простых модулей VBA
- Обзор Visual Basic for Application (VBA)
- Написание макросов VBA
- Изучение объектов VBA
- Технология ActiveX
- Обзор источников по VBA

Какими бы широкими возможностями ни обладали приложения Office XP, они не смогут соответствовать всем требованиям к программному продукту. Если техники работы с мышью, описанной в главе 5, недостаточно, Office предоставляет полноценный инструмент разработки программного обеспечения, называемый VBA. VBA означает Visual Basic для приложений. Он позволяет сделать все, что угодно: от простой (но тем не менее разумной) настройки любого приложения Office XP до создания полноценного программного продукта, использующего возможности всего Office. Не то, чтобы программирование на VBA было таким же простым, как щелканье кнопок на панели инструментов, но это именно то, что каждый может применить для достижения успеха.

Первое знакомство с редактором Visual Basic

На рис. 51.1 показан редактор Visual Basic for Application. Привыкайте к этому окну, поскольку вы будете проводить много времени, работая именно в нем. В качестве поощрения, не придется заново изучать все команды и комбинации клавиш, если захотите работать также и в Visual Basic.

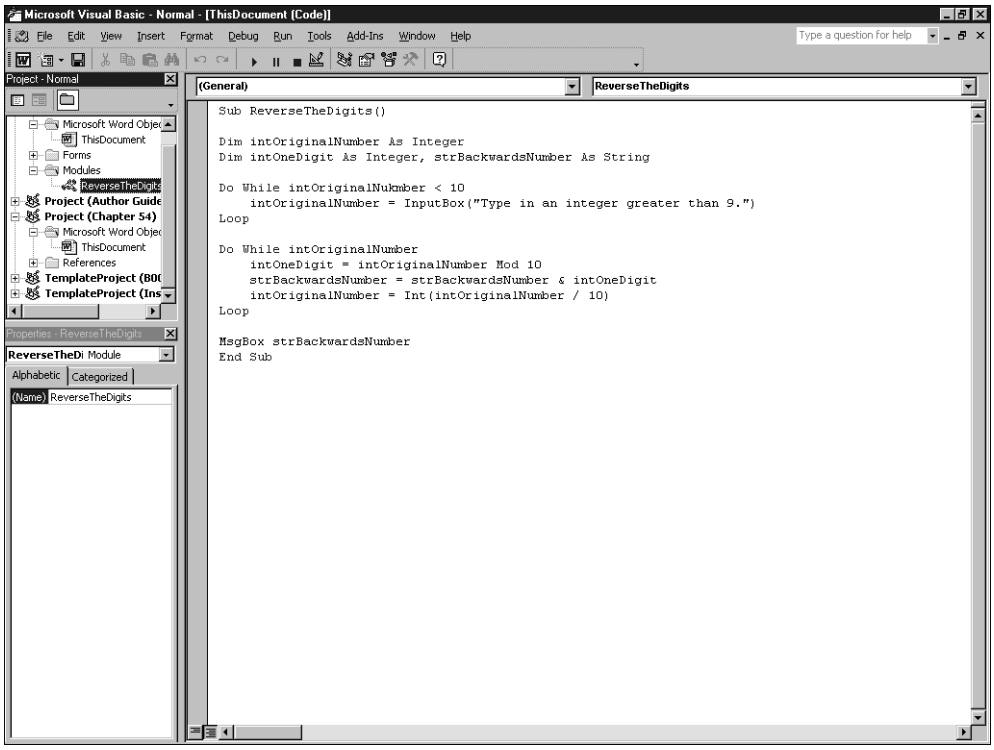


Рис. 51.1 Редактор Visual Basic, который отображает окна проекта, свойств и программы, всплывающее окно справки и панель инструментов VBA

В главе 52 подробно описано использование редактора Visual Basic. На данном этапе просто запомните, что редактор можно запустить только из приложения Office (или другого приложения VBA), но не сам по себе. На экране редактор Visual Basic работает как самостоятельное приложение — в окне полностью независимом от основного приложения. Но поскольку редактор работает с памятью, занимаемой самим приложением, все изменения, вносимые в код программы, сразу же становятся действительными при запуске программы в основном приложении.

Написание простого модуля

Просто для ознакомления с VBA попробуйте написать этот очень простой модуль, состоящий из одной строки. (Возможно, он и “прост”, но довольно полезен.) Для этого выполните следующие действия.

1. Запустите Word.
2. Нажмите <Alt+F11> для запуска редактора Visual Basic.
3. В окне проводника проекта (Project Explorer) выберите для Word шаблон Normal.
4. Чтобы добавить модуль в шаблон Normal, выполните команду Insert⇨Module (Вставка⇨ Модуль). Назовем этот модуль просто: Module1.
5. В окне кода (Code) для Module1 напечатайте `Sub DisplayAppVersion` и нажмите <Enter>. Этим вы указываете VBA, что создаете `Sub` процедуру.
6. Обратите внимание: редактор добавил круглые скобки после введенного имени, а также отдельную строку с инструкцией `End Sub`. Между двумя строками появился мигающий курсор. Теперь введите еще две строки программы, в точности, как они написаны ниже, включая знак подчеркивания в конце первой строки:

```
MsgBox "Versiion/Build: " & application.version &_
    "/" & application.build
```

7. Нажмите <Enter>. Обратите внимание, что при этом VBA автоматически заменяет первые буквы слов *application, version* и *build* на заглавные.

8. Нажмите <F5> — клавишу для запуска модулей в редакторе Visual Basic. Как показано на рис. 51.2, появится сообщение, отображающее номера запущенной версии и копии Word.

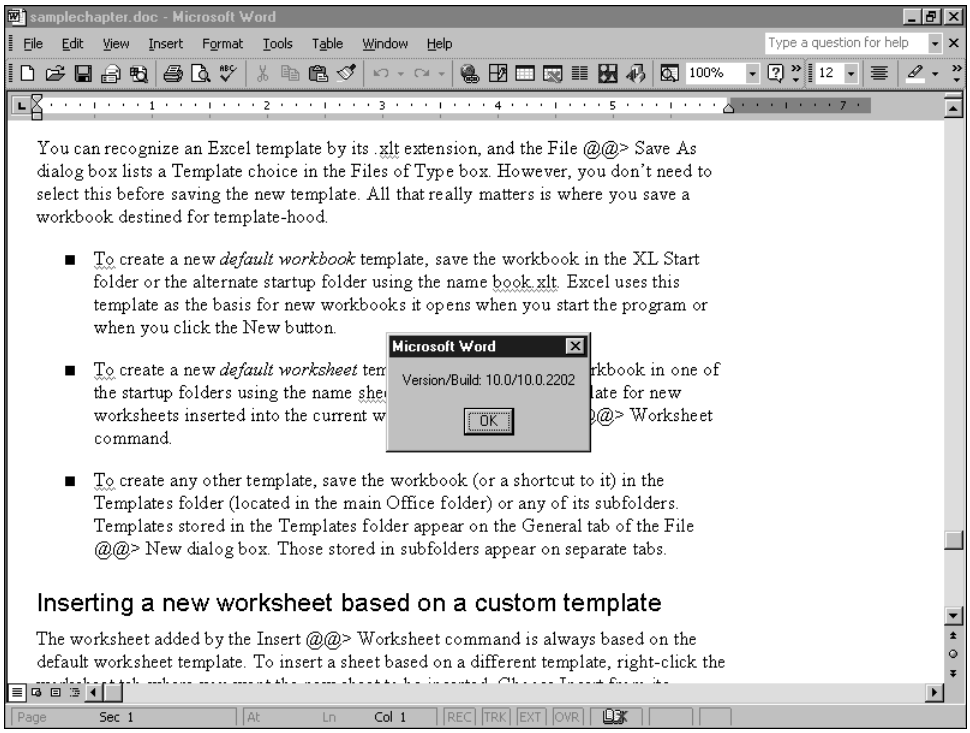


Рис. 51.2. Так выглядит пример программы VBA при запуске (маленькое диалоговое окно посередине окна Windows)

7. Сохраните работу в файле шаблона Normal командой File⇨Save Normal (Файл⇨Сохранить Normal). Потом можете закрыть редактор и вернуться в Word, используя через команду File⇨Close and Return to Microsoft Word (Файл⇨Закрыть и вернуться в Microsoft Word).

Если макрос DisplayAppVersion находится в шаблоне Normal, его можно запустить, выбрав команду Сервис⇨Макрос (Tools⇨Macro) и щелкнув на кнопке Выполнить (Run).

Общие сведение о VBA

Основная часть самого языка программирования, особенностей создания окон и других технологий являются общими для VBA и Microsoft Visual Basic. Visual Basic долгое время был одним из популярнейших инструментов разработки программного обеспечения. Фактически VBA является частью Visual Basic, но имеет свое назначение — позволить каждому расширить функциональные возможности существующих приложений.

Краткий экскурс в историю

VBA, версия Visual Basic "для приложений", была впервые представлена для Excel 5.0. В предыдущих версиях Excel единственной специальной возможностью было написание программы

непосредственно в ячейке особого рабочего листа (эти программы тоже назывались “макросами”, но не были похожими на макросы современного Office). Среда разработки имела ограниченные возможности: сложные программы нельзя было разбить на отдельные блоки, инструменты отладки были слабыми.

С приходом VBA все радикально изменилось. VBA предоставляет возможность разбить программу на логические блоки (процедуры и функции), с которыми можно работать в отдельной программной среде, созданной специально для процесса разработки программного обеспечения. VBA постепенно совершенствовался, появившись в следующем Microsoft Project, а затем в Access. В конце концов, с появлением Office 97 VBA появился в Word и PowerPoint, заменив прежние, ориентированные на приложения, языки Access Basic и Word Basic.

VBA — промышленный стандарт

Использование VBA не ограничивается рамками Microsoft Office. VBA с самого начала предполагался как промышленный стандарт для инструментов программирования и, в конце концов, стал им. Более 200 других разработчиков программного обеспечения купили лицензию на технологию VBA у Microsoft, и VBA появился в таких известных приложениях, как Visio и AutoCAD. Хотя каждое приложение имеет свои программные ресурсы, с которыми необходимо ознакомиться, ядро инструментов VBA — язык и редактор Visual Basic — одни и те же почти во всех приложениях VBA.

С этой точки зрения, VBA и самостоятельная версия Visual Basic тоже почти одинаковы. Главное отличие VBA от Visual Basic заключается в том, что приложение VBA может быть запущено только вместе с основным приложением, а приложения, написанные на Visual Basic, запускаются независимо, как отдельные приложения.

К сожалению, формы и элементы управления VBA отличаются от тех, что есть в Visual Basic, так что при преобразовании приложения VBA в Visual Basic придется разрабатывать новые диалоговые окна и формы. “Плюс” заключается в том, что навыки разработки можно использовать в Visual Basic.

Некоторые важные моменты

Вот список некоторых основных особенностей VBA в Office XP.

- **Все приложения Office содержат VBA.** Такое полезное качество делает программу VBA легко переносимой из одного приложения Office в другое.
- **Все приложения Office имеют достаточно совместимые объектные модели.** Таким образом, каждая из них может работать и как сервер и как контроллер автоматизации (об объектных моделях, серверах и контроллерах речь идет далее в этой главе). Некоторые объекты, такие как `CommandBar` и `Assistant` — общие для всех приложений Office. Сам язык VBA работает как сервер автоматизации, поощряя разработку дополнительных встроенных ресурсов для своей среды программирования.
- **Редактор Visual Basic обеспечивает встроенную среду разработки** (IDE — integrated development environment). Это среда, в которой вы создаете и тестируете модули VBA и формы в отдельных окнах. Редактор Visual Basic — почти идентичный IDE в Visual Basic.
- **Совместимые инструменты разработки окон.** Все полноценные приложения VBA (включая приложения Office, кроме Access) поддерживают единую систему разработки окон (называемых пользовательскими формами (UserForms) в VBA). Преимуществом

являются стандартные наборы инструментов разработки и стандартные элементы управления, которые одинаково работают во всех приложениях Office. Вы можете создать форму, скажем, в Word и использовать ее в Excel *без каких-либо изменений*. Элементы управления, которые поставляются с приложениями VBA, обладают достаточно широкими возможностями для создания сложных форм, имеющих отличный вид (рис. 51.3). Если этого не достаточно, VBA позволяет использовать элементы ActiveX из других источников (см. раздел об элементах ActiveX далее в этой главе).

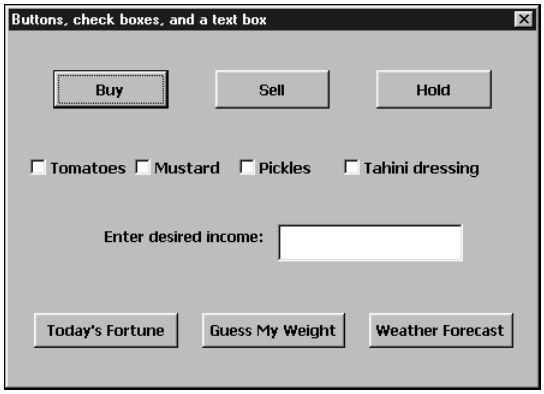


Рис. 51.3. Пример диалогового окна (формы VBA), заполненного элементами ActiveX, которые существуют в VBA

- **VBA позволяет создавать свои объекты, используя модули класса.** Это возможность повторно использовать код программы и сократить время на разработку проекта.
- **ADO (Объекты данных ActiveX) предоставляют документам Office доступ к данным, хранимым в разнообразных источниках**. Стандарт программирования ADO позволяет выполнять поиск и хранение информации, независимо от того находится, она на жестком диске компьютера, сервере рабочей группы, в связанной базе данных или на корпоративной рабочей станции. Помимо однотипных методов для доступа к данным, независимо от их положения, ADO легче в использовании, быстрее и компактнее по сравнению с инструментами доступа к данным в более ранних версиях Office, включая DAO и ODBC Direct.

Подробнее об ADO см. главу 49.

- **Условная компиляция повышает гибкость программы.** Эта возможность позволяет разрабатывать приложения для различных языковых групп пользователей. Условная компиляция также может быть использована для запуска отдельных блоков программы во время отладки.
- **Прямой доступ к Windows API допускает использование Windows в качестве инструмента программирования.** Если вам нужны возможности, которых не может обеспечить VBA, обратитесь к Windows, вызвав Win32 API (интерфейс программирования приложений Windows).

Совместимость VBA? В достаточной мере...

VBA был описан как “достаточно совместимый” во всех приложениях Office. Теперь остановимся на ключевом слове “достаточно”. Несмотря на все попытки Microsoft унифицировать VBA для всех приложений, некоторые несоответствия все же существуют. Вот простой пример. Если вы пишете программу приложения для Word, то на встроенное диалоговое окно Файл⇨Сохранить как (File⇨Save as) вы ссылаетесь как `wdDialogFileSaveAs`, а в Excel на соответствующее окно нужно ссылаться `xlDialogSaveAs`. (Версия Excel не упоминает меню Файл). Более

проблематичным является то, что по сравнению с Word, другие приложения имеют значительно меньше возможностей управления встроенными командами посредством VBA. Будьте готовы бороться с подобными трудностями.

Начинаем работать

VBA является средой программирования, достаточно мощной для ее использования профессиональными разработчиками. Однако даже если вы не обладаете навыками программирования, VBA поможет автоматизировать повторяющиеся задачи в повседневной работе.

Записанные макросы — это программы VBA

Простой способ начать работать с VBA — использовать *средство записи макросов* — инструмент, доступный в Word, Excel и PowerPoint.

Подробнее о механизме записи макроса см. главу 5.

Важным является то, что записанные макросы хранятся как программы VBA, подобно написанным программам. Для изучения программного кода записанного макроса выберите команду Сервис⇨Макросы (Tools⇨Macros), выделите имя нужного макроса и нажмите кнопку Изменить (Edit).

Обратите внимание, что каждый записанный макрос — это одна-единственная процедура VBA. (Если быть более точным, макрос — это `Sub`-процедура, не требующая аргументов.) По умолчанию Office хранит все записанные макросы как отдельные процедуры в одном модуле (единице программы, которая может содержать одну или более процедур) текущего документа.

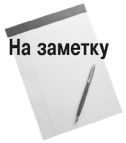
Процедуры и модули полностью освещены в главе 54.

Пределы возможностей макросов

Макросы применяют, чтобы ускорить выполнение последовательности команд, которые используются в одном и том же порядке. Проблема с записанным макросом заключается в том, что он слишком “примитивен” — всегда выполняет те же действия в той же последовательности. Чтобы создать программу, которая может изменять поведение в зависимости от текущих условий, ее нужно вручную написать на VBA.

Вот еще несколько причин для перехода от средства записи макросов к VBA.

- **Создание пользовательских диалоговых окон и форм.** VBA позволяет конструировать как простые окна сообщений типа “Да-Нет-Отмена”, так и сложные интерактивные окна со множеством вкладок — и все это без программирования. VBA — визуальный инструмент разработки программного обеспечения. Вы конструируете формы на экране, просто “рисуя” их мышью.



Чтобы формы выполняли полезную работу, программированием все же придется заняться.

- **Поиск данных в файловых базах данных, таких как Access, dBase или FoxPro, а также во встроенной системе управления реляционными базами данных посредством OLE DB или ODBC.** Для просмотра данных из этих источников можно использовать компонент Microsoft Query, но чтобы манипулировать ими в своей программе, необходимо написать программный код VBA.
- **Управление ошибками в подпрограммах VBA.** Даже средство записи макросов может выиграть от небольшой процедуры, которая выполняется при возникновении ошибки, позволяя подпрограмме "мягко приземлиться". Процедуры обработки ошибок особенно важны при распространении программы VBA среди других пользователей, запускающих программное обеспечение в условиях, которыми вы не сможете управлять. VBA предлагает инструмент отлова ошибок с широкими возможностями, сравнимый с используемым в Visual Basic.
- **Создание специальных сообщений справки.** При помощи VBA в зависимости от ситуации можно добавить окно справки к программе и отобразить соответствующее сообщение, когда пользователь допускает ошибку. Можно даже создать Помощника по Office, чтобы провести пользователя через какой-либо сложный процесс.
- **Защита от несанкционированного доступа и копирования.** Использование пароля позволит защитить код программы от просмотра посторонними.

VBA: инструмент разработки, основанный на объектах

Хотя VBA не удовлетворяет всем критериям настоящего объектно-ориентированного языка программирования, объекты играют в нем фундаментальную роль. Поэтому для эффективного использования VBA необходимо понять, как работают объекты.

Подробнее объекты VBA рассматриваются в главе 54.

Суть объектов VBA

Если вы не знакомы с объектно-ориентированным программированием, самым простым способом понять, что такое объекты является следующий: представьте их частями основного приложения и создаваемых в нем документов. В Excel, например, каждая ячейка рабочего листа — это объект, каждый именованный диапазон ячеек — объект, диаграмма — объект, весь рабочий лист — объект и вся рабочая книга — тоже объект.

Но объекты могут быть и более абстрактными. В том же Excel объект `CustomView` является пользовательским видом рабочей книги. Пользовательский вид — не то, что вы можете увидеть на экране. Это набор установок, определяющих вид рабочей книги, а также параметры печати. Во всех приложениях VBA объект `Collection` представляет любую группу переменных или других объектов, которые работают как единое целое, независимо от их типа.

Работа с объектными моделями

Каждый объект VBA занимает определенное место в определенной основным приложением и самим VBA иерархии других объектов. Объект, который находится на вершине иерархии, имеет особые свойства, методы и события. Он также служит контейнером для других типов объектов (тех,

что находятся ниже по иерархии). Эти объекты содержат другие подчиненные объекты и так далее. *Объектная модель* приложения VBA наряду с характеристиками отдельных объектов отражает специфику иерархических отношений между объектами.

Так как сам VBA имеет объекты, доступные во всех приложениях VBA, то у него тоже имеется своя объектная модель.

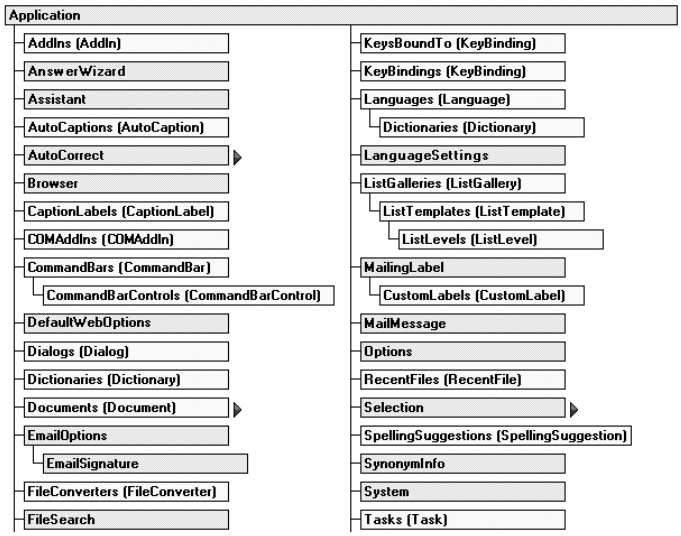


Рис. 51.4. На рисунке представлена часть схемы объектной модели Word

Вверху объектной иерархии находится объект `Application`. Под объектом `Application` — производные семейства (библиотеки) объектов, такие как `Documents`, `Paragraphs`, `Characters`, `Bookmarks`, `Dialogs` и другие.

Технология ActiveX и VBA

ActiveX — торговый термин, объединяющий разнообразные технологии работы со взаимозаменяемыми программными компонентами. Отдельные технологии ActiveX в действительности существенно отличаются, но их можно рассматривать вместе как набор стандартов для составных частей программного обеспечения. Эти составные части зачастую смешиваются и сочетаются для создания мощного пользовательского приложения.

Элементы ActiveX

Элементы управления — это отдельные объекты на панелях инструментов и диалоговых окнах, которые пользователь может нажать или ввести в них текст для обработки программным обеспечением. Примеры элементов управления — кнопки, переключатели для выбора одного из нескольких заданных параметров, поля, в которых можно вводить или редактировать значения.

Стандарт Microsoft *элементов ActiveX* не ограничен элементами управления VBA. Элементы ActiveX являются взаимозаменяемыми программными механизмами, которые можно использовать в программе, и они будут работать в Visual Basic, C++, Java так же, как и в VBA. Если нужен элемент управления, которого нет в VBA, существуют сотни других доступных для совместного использования элементов ActiveX от сторонних разработчиков программного обеспечения. Вы можете создать и свои элементы ActiveX (используя Control Creation Edition для Visual Basic), если захотите.

Хотя элементы управления обычно считают элементами, которые видны в диалоговом окне или на панели инструментов, некоторые элементы ActiveX не имеют визуального представления. Вместо этого, они обеспечивают программную функциональность, предоставляя такие специальные возможности, как денежные вычисления.

Элементы ActiveX могут переноситься с одного компьютера на другой или с одной операционной системы на другую, когда используются в Web. Подробнее см. главу 56.

В Web-обозревателе, поддерживающем стандарт ActiveX, пользователи могут и работать с элементами ActiveX, которые появляются на странице. Следовательно, технология ActiveX распространяет программную модель компонента за границы, создаваемые различными операционными системами и сетевыми топологиями.

Автоматизация ActiveX

Программы VBA не ограничиваются использованием объектов основного приложения. Наоборот, вы имеете доступ к объектам любого приложения или программного компонента, поддерживающим стандарт Component Object Model (COM). Спецификация COM описывает, как объекты должны быть определены и объявлены в приложении, чтобы они могли использоваться другими приложениями. Термин *автоматизация* (или *программирование объектов*) относится к возможности приложений, которые поддерживают стандарт COM, активизировать и управлять объектами других приложений.

Автоматизация и COM открывают фантастические возможности для специальных приложений VBA, впечатляющих своими масштабами. Если вы действительно амбициозны, то можете создать одно приложение, которое будет комбинировать и работать с информацией из документов Word, рабочих листов Excel, баз данных Access и Outlook, презентаций PowerPoint, схем Visio, технических схем AutoCAD и т.д. Программное решение зачастую отображает всю эту информацию в специальных формах и в окнах отдельных приложений, автоматизированных вашей программой. Используя технологию DCOM (Distributed Component Object Model), компоненты ActiveX могут распространяться и поддерживаться в сети, независимо от того, является она сетью LAN или Intranet.

Серверы и контроллеры автоматизации ActiveX

Приложения, которые предоставляют собственные объекты для использования другими программами, называются *серверами автоматизации ActiveX* (или *серверами программирования ActiveX*). Приложения, которые могут использовать объекты другого приложения, называются *контроллерами автоматизации ActiveX* (или *контроллерами программирования ActiveX*). Каждое приложение в Office может выступать и в роли сервера, и в роли контроллера.

Источники информации о VBA

Если введение в курс VBA, размещенное в этой книге, вызвало у вас желание узнать больше, начните с “VBA для чайников”, второе издание (Hungry Minds, Inc.) Стива Каммингса (Steve Cummings). Хотя представленный материал имеет начальный уровень, в нем можно найти важные детали.

Источники для программистов VBA существуют в большом количестве в Internet. Начнем с начала, с Web-страничек Microsoft по VBA и Visual Basic:

- `msdn.microsoft.com/vba`
- `msdn.microsoft.com/vbasic`

Также можно найти много информации по VBA на `www.microsoft.com`, в разделах о конкретных приложениях Office и в Microsoft Knowledge Base.

Можно исследовать и другие страницы Web:

- `www.download.com/PC/Activex`
- `www.geocities.com/Wallstreet/9245/`
- `www.cgvb.com/`
- `www.mvps.org/vbnet/`
- `www.vbip.com`
- `http://searchvb.techtarget.com/`
- `www.codehound.com/vb/`

Периодические издания по VBA:

- *Microsoft Developers Network (MSDN),* Web-страница `http://msdn.microsoft.com`
- *Element K Journals,* тел: 800-223-8720, Web-страница `www.elementkjournals.com`
- *Microsoft Office & Visual Basic for Applications Developer*, Informant Communications Group, Inc., 10519 E. Stockton Blvd., Suite 100, Elk Grove, CA 95624-9703; тел: 916-686-6610, Web-страница `www.informant.com`
- *Visual Basic Programmer's Journal*, Fawcette Technical Publications, Inc., 913 Emerson Ave., Palo Alto, CA 94301; 650-566-200, Web-страница `www.windx.com`