Глава 57

Создание функциональных приложений

В этой главе...

- Построение приложений с использованием объектов COM
- Создание надстроек для определенных приложений
- Повышение универсальности программ за счет объявления свойств документов в качестве переменных
- Обеспечение безопасности и блокировка доступа к готовому коду
- Расширение созданных приложений при помощи Office Developer
- Распространение готового приложения

Если вы используете VBA для написания отдельных модулей, чтобы сделать свою работу в Office более продуктивной — это хорошо и здорово. Однако VBA — полнофункциональное средство разработки программного обеспечения, его можно применять для создания готовых приложений, которые помогут в решении практически всех возникающих задач. По функциональности ваше приложение VBA может быть равно любой комбинации приложений Office.

Построение приложений с объектами COM

Для многих приложений VBA основной задачей является решить, расширить или, напротив, сосредоточить на чем-нибудь функциональность стандартного приложения Office. Например, в таблицу Excel можно добавить набор специальных финансовых и отчетных функций, которые помогут составить месячный отчет. В дальнейшем целесообразно дополнить пользовательский интерфейс Excel при помощи новых элементов управления и панелей инструментов. А тем, кто не желает разбираться со структурой меню Excel, рекомендуем использовать кнопочный интерфейс. На рис. 57.1 показано приложение на базе Excel.

Рис. 57.1. Приложение VBA, использующее кнопочный пользовательский интерфейс для менее опытных пользователей. Обратите внимание на панели управления в верхней части окна приложения

Мир программного обеспечения простирается далеко за границы отдельного приложения VBA, однако VBA позволяет путешествовать везде. Все приложения Office и дополнительные компоненты, включенные в модель компонентных объектов, — COM (Component Object Model) — рекомендуется применять в своих программах. Это означает, что можно строить приложения, использующие возможности Word, Excel и Access. Таким образом, приложения VBA могут превосходить по возможностям любое приложение на основе COM, независимо от того, входит оно в Office или нет. Создать организационную диаграмму Visio, основанную на личных данных, хранимых в папках Outlook, можно при помощи VBA. На рис. 57.2 показана форма Outlook, основанная на таблице Excel.

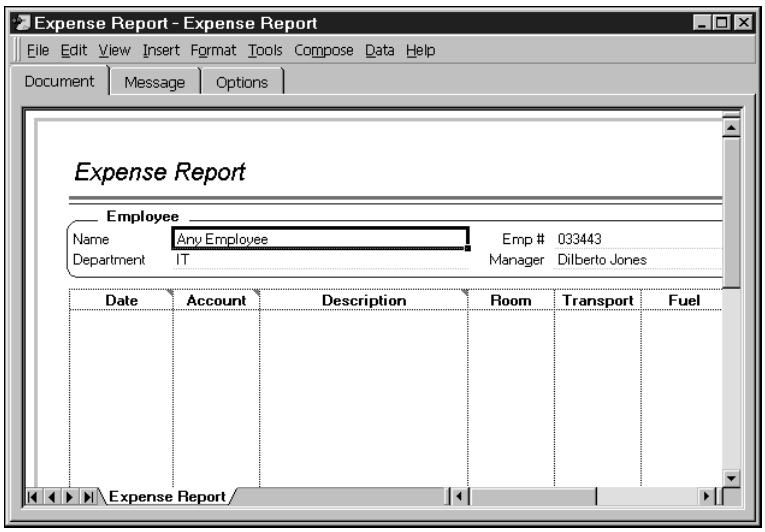


Рис. 57.2. Форма Outlook, основанная на данных таблицы Excel

COM учит разработчиков тому, что большое программное обеспечение создается из уже использовавшихся частей — заново изобретать велосипед для каждого

приложения не очень умно, это лишняя трата времени и сил. Только подумайте о том, сколько функциональности делает доступным Office. Access великолепно работает в качестве базы данных, с первичными ключами и индексами таблицы. Excel поддерживает сложные статистические и финансовые формулы, которые тяжело воспроизвести в Access. Word предоставляет возможности по редактированию и форматированию текста, намного превосходящие возможности Access и Excel. А при помощи COM можно совмещать эти программные компоненты как угодно, чтобы достичь нужных целей при проектировании.

В коде, приведенном в этом разделе, хранимые в Outlook данные, извлекаются и передаются для анализа в Excel. Пример кода демонстрирует множество концепций, которые необходимо освоить для построения VBA-приложений.

Задание

Вы мечтаете об отпуске на Гавайях, который планировали еще за год. Вы и ваша команда оканчиваете программный проект, над которым работали несколько месяцев по 12 часов в сутки, и отдых действительно вам необходим. Через неделю вы будете нежиться в теплых водах Тихого Океана! В 16:30 начальник отдела просит вас зайти на минутку… И сообщает, что высшее руководство поставило задачу, требующую немедленного решения.

Похоже, что руководство вложило средства в приложение для рабочих групп для работы с общими папками сервера Excange, которое отслеживало бы контакты с потребителями, маркетинговые исследования и результаты продаж в суммах общих продаж. Премии отдельных работников маркетингового отдела зависят от данных, которые необходимо выбрать из общей папки контактов с потребителями, проанализировать в Excel и представить руководству для принятия окончательного решения в тот самый день, когда вы должны ехать в отпуск.

Определение проблемы

Перед тем как написать хотя бы строчку кода, необходимо максимально точно определить решаемую задачу. Пока цель не будет четко ясна, ничего толкового не выйдет. Приняв это к сведению, набросайте письмо начальнику с запросом дополнительной информации. Прочитав ответ, вы поймете, что никто не требует невозможного; просто необходимо отобрать информацию так, чтобы гуру по таблицам из бухгалтерии могли совершать над ней свои заклинания. Таким образом, задача сводится к переносу данных из одного контейнера в другой.

Когда весь проект можно свести в конкретные рамки, следующий этап — разбить его на части, отвечающие за выполнение тех задач, которые необходимы для достижения конкретной цели. Затем при помощи программного обеспечения для построения диаграмм типа Visio или при помощи карандаша и бумаги нужно набросать структуру приложения.

Предположим, приложение запрашивает пользователя, финансового аналитика, диапазон дат и категорию потребителей, а затем извлекает данные из папки Outlook в таблицу. Общая цель заключается в перемещении данных из одного контейнера (папки Outlook) в другой контейнер (таблицу Excel).

К счастью, оба приложения являются компонентами Office. Office построен на технологии ActiveX, которая в своей основе содержит концепцию многократно используемых компонентов, определяемых как объекты со свойствами, событиями и методами. Outlook позволяет получить доступ к объектной модели таким приложениям VBA, в которых хранится код, например Excel.

Итак, задача сводится к тому, как определить объекты Outlook, получить доступ к свойствам этих объектов (их данным) и разместить полученные свойства в объектах Excel (листах, строках, столбцах). Когда условие задачи определено в терминах ActiveX, ее решение превращается в потрясающее приключение, а не в прямой путь к бессоннице, как казалось на первый взгляд.

Начало

Начните новый проект VBA, запустив ведущее приложение Office, в котором будет храниться проект, и открыв в этом приложении новый документ. (В примере кода, представленного далее в главе, ведущим приложением является Excel.) Быстро запустить редактор Visual Basic можно при помощи комбинации клавиш <Alt+F11>.

Программирование при работе с несколькими приложениями

Отметим одну (возможно, очевидную) деталь: использовать объекты другого приложения можно, только если это приложение установлено в системе. Если это так, работа с другим приложением на основе COM требует следующих предварительных шагов.

1. В редакторе Visual Basic добавьте ссылку на библиотеку объектов внешнего приложения.
2. Объявите переменные для объектов, которые планируется использовать в программе.
3. Создайте экземпляр объектов, используя функцию `CreateObject`.

В следующих трех разделах о каждом из этих шагов рассказано подробнее (о технике работы с объектами в VBA-коде см. главу 49).

Добавление ссылки на новую объектную модель

Чтобы сообщить редактору Visual Basic об объектной модели внешнего приложения, добавьте и активизируйте ссылку на объектную модель приложения. В редакторе Visual Basic, выберите команду Tools⇨References (Сервис⇨Ссылки) с целью открыть диалоговое окно References (Ссылки) (рис. 57.3).

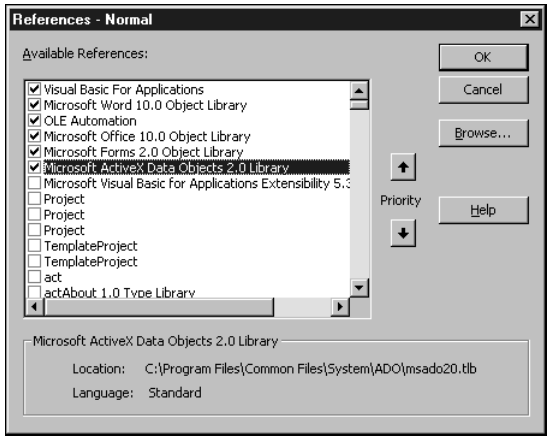


Рис. 57.3. В этом окне открывают и активизируют ссылки на библиотеки объектов

Библиотека объектов Outlook должна присутствовать в диалоговом окне (если, конечно, Outlook уже установлен на компьютере). Достаточно найти ее в списке Available References (Доступные ссылки) и установить флажок, чтобы ее активизировать и сделать ее объекты доступными. Если же нужная библиотека объектов отсутствует, добавьте ссылку в список самостоятельно при помощи кнопки Browse (Обзор).

Объявление внешних переменных

Чтобы объявить переменные для объектов из внешних приложений, используйте стандартный синтаксис VBA. Узнать, какие объекты доступны, можно, использовав средство просмотра объектов (Object Browser) или файл справки внешнего приложения. Следующие операторы в разделе объявлений (Declarations) модуля объявляют переменные для необходимых программе объектов Outlook:

```
Dim objOutlook As Outlook.Application
Dim objOLNamespace As Outlook.NameSpace
Dim colFolders As Outlook.Folders ' библиотека папок
Dim objPeopleFolder As Outlook MAPIFolder

Dim colPeople As Outlook.Items ' библиотека контактов
Dim objPerson As Object ' один личный контакт
Dim strName As String
```

Создание внешнего объекта

Объявление объектов просто показывает намерение их использовать. Затем нужно создать объекты при помощи операторов `Set` в процедуре VBA.

Ключом к созданию объекта, принадлежащего внешнему приложению, является функция `CreateObject`. `CreateObject` запускает приложение, которое, в свою очередь, создает объект. Возвращаемое значение функции — ссылка на объект, которую можно присвоить соответствующей переменной. Чтобы запустить приложение с определенным файлом или документом, используйте функцию `GetObject`.

Ниже приведен пример процедуры для пары Excel–Outlook, которая создает экземпляры объектов:

```
Sub PeopleWorksheet()

' запускает Outlook, создает объектную ссылку на приложение
Set objOutlook = CreateObject("Outlook.Application.10")

' создает ссылки на объекты Outlook
Set objOLNamespace = objOutlook.GetNamespace("MAPI")
Set colFolders = objOLNamespace.Folders ' все папки
Set objPeopleFolder = colFolders.Item("Personal Folders")

' углубляемся в иерархию папок
' заново используем некоторые имена переменных — предыдущие
объекты больше не нужны
Set colFolders = objPeopleFolder.Folders
Set objPeopleFolder = colFolders.Item("Customer Contacts")
Set colPeople = objPeopleFolder.Items

' вызов процедуры Sub, которая использует данные Outlook:
OutlookToExcel
End Sub
```

Пока процедура `OutlookToExcel` не определена, при выполнении примера `PeopleWorkSheet ()` будет генерироваться ошибка. Кроме того, иерархия папок приложения Outlook, установленного на компьютере, должна содержать папки “Personal Folders” и “Customer Contacts”.

Во многих ситуациях, включая данный пример, функция `CreateObject` используется только однажды. Когда объект внешнего приложения создан, можно получить доступ к его свойствам и методам. В данном случае, как обычно, эти свойства включают в себя другие объекты (здесь — библиотека `Folders`, папки `Personal` и папки `Contacts` и библиотека элементов папок `Contacts`).

Приложения, которые запускаются из кода при помощи функции `CreateObject`, работают в скрытом режиме — их не видно на экране, что полезно, когда нужно использовать данные приложения, не отвлекая внимание пользователя.

Однако иногда необходимо увидеть приложение воочию. В зависимости от приложения, можно активизировать метод `Display` или установить значение свойства `Visible` равным `True`, применив код следующего вида:

```
objPeopleFolder.Display
```

Использование внешних объектов

Теперь можно переходить к использованию в коде объектов внешних приложений, подобно тому, как если бы они были объектами приложения VBA:

```
Sub OutlookToExcel()
For each objPerson in colPeople
    With objPerson
' передает данные из элемента контакта в переменные
        strName = .FullName
        curSales = .SalesTotal
        ...(здесь должен быть код, который вставит эти данные в
таблицу и проанализирует их)
    End With
Next
End Sub
```

Гавайи, я лечу к вам!!!

Обратите внимание: в этом примере использование объектов Outlook упрощает получение данных, хранимых Outlook. Однако, поскольку в коде используется доступ к объектам другой программы, целесообразно запустить те действия, которые они могут выполнить. Вам разрешено открывать, редактировать и сохранять документы в “чужом” приложении. А можете использовать инструменты анализа данных, даже если эти данные из VBA-кода, а не из другого приложения.

Построение надстроек

VBA-код, выполняемый в обычном документе, хорош для личного использования, но он не является удачным решением для продуктов, предназначенных к распространению. Создав собственные надстройки, можно спокойно объединить свои

наработки с приложением Office, создав у пользователя стойкое впечатление, что они являются неотъемлемой частью изначальной программы.

В Office надстройка — просто программный компонент, добавляющий функциональность к одному или нескольким приложениям Office. Но в отличие от обычных документов, содержащих код VBA, надстройки имеют две особенности.

- Надстройки доступны из любого документа.
- Когда надстройка загружена, ассоциированные с ней документы недоступны пользователю. Надстройка может вносить изменения в пользовательский интерфейс приложения, но единственный способ убедиться, что она запущена — проверить соответствующее диалоговое окно (обычно Сервис⇨Надстройки (Tools⇨Add-Ins)).

Создание надстроек под определенные приложения

В версиях, предшествовавших Office 2000, отдельная надстройка могла работать только с одним приложением Office и, в большинстве случаев, состояла из обычного VBA-кода, который компилировался каждый раз при ее запуске. Сейчас тоже можно создавать такие надстройки, и достаточно просто. Далее изложены основные этапы.

1. Создайте для надстройки новый документ.
2. В новый документ добавьте VBA-модуль и формы, напишите код и оттестируйте программу. В Access также необходимо добавить специальную таблицу, содержащую такую информацию из реестра, которая позволяет устанавливать надстройку.
3. Сохраните документ как надстройку, соответствующую данному приложению (в Word как шаблон `.dot`, в Excel как `.xla`, в PowerPoint как презентацию `.ppt` и в Access как надстройку `.mda` или как базу данных `.mde`).
4. Установите надстройку, используя диалоговое окно Сервис⇨Надстройки. (В Word выберите команду Сервис⇨Шаблоны и надстройки (Tools⇨Templates and Add-Ins) и щелкните на кнопке Добавить (Add), чтобы загрузить шаблон как надстройку. В Access, выберите команду Сервис⇨ Надстройки⇨Диспетчер надстроек (Tools⇨Add-Ins⇨Add-In Manager).)

Создание надстроек COM

В Office 2000 представлены надстройки COM — новый и более мощный тип надстроек. Надстройка COM — это скомпилированная динамически присоединяемая библиотека (DLL), содержащая специальные методы, которые позволяют ее загружать в приложениях Office. Исходя из названия, такая надстройка работает с помощью объектной модели нужного приложения Office через технологию COM и обладает следующими преимуществами.

- Поскольку надстройки COM уже скомпилированы, они загружаются и работают быстрее, чем аналогичные надстройки, написанные старым способом.
- Одна надстройка COM может при необходимости выполняться в нескольких приложениях Office.

- Надстройки COM работают в Outlook и FrontPage, которые не поддерживают надстроек, написанных под определенные приложения.

Можно создать надстройку COM, используя разные средства разработки Microsoft. VBA, конечно, тоже работает, но только при наличии Office XP Developer (см. раздел "Использование Office Developer" далее в этой главе). В Developer входит такое специальное программное обеспечение для компиляции кода надстроек, как DLL, и регистрация DLL в Windows, а также полное руководство по созданию надстроек COM с использованием Visual Basic и VBA. При желании, для создания надстроек COM можно воспользоваться Visual C++ или Visual J++.

Использование настраиваемых свойств как переменных документа

Часто при программировании возникает ситуация, когда необходимо сохранять значение переменных, даже если программа не запущена. Возьмем, к примеру, обычный счетчик. Предположим, необходимо узнать, сколько раз в течение месяца пользователь щелкает на кнопках панелей инструментов Excel. Если, конечно, не держать Excel открытым весь месяц, нужно где-то сохранять значение счетчика каждый раз при выходе из программы и считывать обратно при запуске.

Среди всех приложений Office только Word содержит в объектной модели инструменты, необходимые для прямого управления постоянными переменными (через объект `Variable`). Однако стандартные свойства документа обеспечивают обходной путь, отвечающий всем требованиям Excel и PowerPoint. Можно использовать VBA для создания настраиваемых свойств документа, заполнить их данными и считать данные по требованию. Поскольку данные будут являться неотъемлемой частью программы, не придется беспокоиться о сохранении их в отдельном файле — при сохранении документа его свойства будут автоматически сохраняться на диск вместе со всем его содержимым.

Подробнее о свойствах документа и работе с ними см. главу 12.

Чтобы создать новое свойство документа в Word, Excel или PowerPoint, используйте метод `Use` библиотеки документа `CustomDocumentProperties`. Следующий пример работает в Excel, где объект `ActiveWorkbook` ссылается на любую рабочую книгу, активную во время запуска процедуры (похожие объекты в Word и PowerPoint — `ActiveDocument` и `ActivePresentation`, соответственно):

```
ActiveWorkbook.CustomDocumentProperties.Add _
    Name:= "Button Count", LinkToContent:=False, _
    Type:=msoPropertyTypeNumber, Value:=0
```

Параметр `LinkToContent` должен быть установлен равным `False`, если не нужно, чтобы значение свойства было основано на диапазоне или закладке. Параметр `Type` определяет тип данных свойства; возможны следующие константы: `msoPropertyTypeBoolean`, `msoPropertyTypeDate`, `msoPropertyTypeFloat` и `msoPropertyTypeString`.

Чтобы получить информацию, которая хранится в свойстве документа, воспользуйтесь свойством объекта `Value` с таким оператором, как:

```
intCurrentCount = _
    ActiveWorkBook.CustomDocumentProperties("Button
Count").Value
```

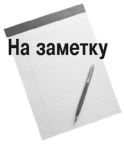
Существуют, по меньшей мере, два альтернативных решения данной проблемы. Можно записать и считывать данные из реестра Windows, используя операторы `SaveSetting` и `GetSetting`, либо сохранить данные в отдельный файл и считывать их оттуда при помощи операторов `Put` и `Get`.

Обеспечение безопасности кода при помощи цифровых сертификатов

Как и все достаточно мощное в этом мире, язык VBA может быть использован в разных целях. Процедуры VBA могут содержать вирусы, которые сеют в системе хаос и опустошение. По этой причине код, написанный неопытным программистом, пусть даже и с хорошими намерениями, иногда причиняет серьезные проблемы.

Чтобы уменьшить такую угрозу, VBA позволяет защищать проекты, используя цифровые сертификаты. Используя аналогию Microsoft, цифровые сертификаты выполняют ту же функцию, что и сургучная печать на конверте, которая позволяет получателю определить кто отправил проект и изменялось ли содержимое (намеренно или ненамеренно). В отличие от печати, цифровой сертификат не гарантирует, что никто не заглядывал в проект. (Чтобы другие не разглядывали код, следуйте инструкциям, изложенным в разделе “Блокировка доступа к коду” далее в этой главе.)

Пользователь, который открывает файл или загружает приложение, подписанное цифровым сертификатом, видит цифровую подпись, содержащую имя человека, подписавшего сообщения и другую идентификационную информацию. Если все в порядке, пользователь принимает и выполняет код. Если же цифровая подпись отсутствует или идентификационная информация неверна, пользователь должен самостоятельно решить, нужно ли открывать или загружать элемент.



Цифровые подписи распознаются, только если в системе установлен броузер Internet Explorer версии 4.0 или выше.

Прежде, чем подписывать работу, необходимо получить и установить на свой компьютер один или несколько цифровых сертификатов. Цифровые сертификаты распространяются организациями, например VeriSign, к которым разработчик может обратиться как частное или юридическое лицо. Если вы являетесь штатным программистом, то, скорее всего, не обладаете полномочиями требовать цифровой сертификат для всей компании. Организация получает сертификаты и устанавливает систему для работы с ними, и запрос осуществляется через человека, который отвечает за безопасность.

Как бы там ни было, когда на компьютере цифровой сертификат установлен, цифровая подпись текущего проекта будет естественным продолжением. В редакторе Visual Basic выберите команду Tools⇨Digital Signature (Сервис⇨Цифровая подпись). В диалоговом окне Digital Signature (Сертификат) щелкните на кнопке Choose (Выбрать), чтобы появился список сертификатов, установленных на компьютере. Для активизации подписи выберите нужный сертификат и щелкните на кнопке OK в обоих диалоговых окнах.

Если проект содержит цифровую подпись, любое изменение, внесенное в проект при отсутствии оригинального сертификата, повреждает и снимает подпись. Если планируется передать подписанный проект кому-либо, не изменяйте его, пока на компьютере нет копии цифрового сертификата.

Во многих организациях проекты VBA, на самом деле, *подписывают* только менеджеры проекта. Разработчик выполняет проект и передает его на проверку человеку, который ставит подпись. Только затем проект попадает к пользователям. (Хотя, конечно, не помешает знать, как подписывается проект, хотя бы на случай, если ответственный имеет полномочия, но не знает, как это делается.)

Блокировка доступа к коду

Если вы разрабатываете приложение для продажи и не желаете пускать других к плодам своего тяжкого труда, необходимо защитить созданный код, спрятав его так, чтобы никто не мог его увидеть или изменить. Защита кода имеет смысл, если код распространяется в общей среде и требует защиты от необдуманных или преднамеренных изменений. После того, как проект отлажен и оттестирован, для защиты кода выполните следующие шаги.

1. Выберите команду Tools⇨<*Имя_проекта*>Properties (Сервис⇨Свойства:-<*Имя_проекта*>) и в диалоговом окне Project Properties (Свойства проекта) перейдите на вкладку Protection (Защита).
2. Установите флажок Lock Project for Viewing (Заблокировать просмотр), чтобы никто не мог видеть код.
3. Дважды введите пароль, чтобы подтвердить правильность ввода.

Не ошибитесь с паролем, иначе в дальнейшем изменить код не удастся.

4. Щелкните на кнопке OK.

Если доступ к проекту заблокирован, в окне Project (Проект) видно только его имя. Любому, кто захочет увидеть содержимое проекта, придется сначала ввести правильный пароль.

Можно ввести пароль, и не устанавливая флажок Lock Project for Viewing. В этом случае пароль потребуется для открытия диалогового окна Project Properties, а не для просмотра или изменения кода или форм проекта.

Использование Office Developer

Для тех, кто занимается серьезной работой по разработке приложений VBA, Office XP Developer — обязательное приложение. Office XP Developer — это специальный выпуск Office, ориентированный на программистов VBA. В табл. 57.1 перечислены его инструменты и другие компоненты, входящие в комплект.

Таблица 57.1. Инструменты Office XP Developer

Инструмент	*Функция*
Package and Deployment Wizard	Создает установочные диски для распространения приложений
Visual Source Safe	Отслеживает версии
COM-Add-Ins Desiner и шаблоны	Преобразует VBA-код в DLL-библиотеки надстроек; шаблоны позволяют создавать приложения COM с использованием Visual Basic, Java или C++
Code Librarian	Предоставляет возможность хранить и повторно использовать код базы данных, оснащенной системой поиска и включающей большой объем готового кода на разных языках Microsoft
Error Handler	Автоматизирует добавление в код стандартизованных процедур обработки ошибок
Code Commenter	Автоматически добавляет блоки комментариев в код согласно спецификации
String Editor	Помогает в написании операторов SQL для доступа и работы с базами данных
Data Environment Designer	Обеспечивает более легкое соединение с базами данных ведущих приложений (но не Access)
Data Report Designer	Позволяет проектировать отчеты баз данных, отличных от Access
Дополнительные элементы управления ActiveX	Включают в себя информационные элементы управления и элементы Common Dialog
HTML Help Workshop	Позволяет создавать системы HTML-справки для приложений

Access Run-time	Разрешает распространять приложения Access пользователям, у которых нет самого Access
Replication Manager	Отображает и управляет копированием баз данных Jet в сетях или Internet

Распространение решений

Чтобы приготовить решение к распространению другим пользователям, обратитесь к мастеру упаковки и размещения, включенному в Office Developer. Этот мастер создает файлы, необходимые для установки и запуска приложения. Полный обзор мастера упаковки и размещения не рассматриваются в этой главе. Однако остановимся на некоторых этапах его деятельности.

- Определяет файлы, которые необходимы приложению, включая основной файл приложения, содержащий объекты, и код приложения; файлы данных, используемых приложением; пиктограммы, рисунки, звуковые или мультимедийные файлы; стандартные файлы справки (включая элементы управления ActiveX, DLL и файлы периода выполнения).
- Создает ярлыки и элементы меню Пуск (Start) в Windows.
- Записывает ключи реестра Windows (или изменение существующих ключей), необходимые для работы приложения.
- Определяет перераспределенные компоненты приложения, например ODBC.
- Определяет параметры обычной, компактной установки или установки по выбору.
- Устанавливает параметры носителей (дискеты, сеть, CD-ROM).

Построение самостоятельных приложений Access

При разработке приложений с использованием Excel, Word, PowerPoint или Outlook помните, что эти же элементы Office должны быть установлены у конечных пользователей. Ситуация с приложениями Access иная, но при условии наличия Office Developer. Распространяя со своим приложением модуль Access Run Time (компонент Office Developer), разработчик обеспечивает запуск приложения на любом компьютере, независимо от того, установлено ли там приложение Access. Разработчик может свободно распространять Jet или MSDE.