УРОК

Процедуры, модули и модули классов

Содержание урока

- Понятие процедуры
- Работа с модулями
- Использование модулей классов
- Создание и выполнение процедур

В предыдущих уроках было продемонстрировано, как процедуры событий могут расширить возможности приложения Access. Процедуры событий — это только один из нескольких доступных типов процедур. Процедуры событий выполняются автоматически в ответ на определенное действие в форме или отчете, а подпроцедуры и функции обеспечивают еще одни способ обработки действий в приложении.

Понятие подпрограммы

Подпроцедура, или подпрограмма (*subprocedure*) — это ряд операторов в программе, которые выполняют некоторое действие. Подпроцедуры начинаются с ключевых слов `Public Sub` и заканчиваются оператором `End Sub`. Они могут получать аргументы, но не возвращают значение. Ниже приведен пример подпроцедуры.

```
Public Sub IsWeekday(dtmDateToCheck As Date)
   If Weekday(dtmDateToCheck) = vbSaturday Or _
     Weekday(dtmDateToCheck) = vbSynday Then
     MsgBox dtmDateToCheck & " — выходной день.", vbInformation
   Else
     MsgBox dtmDateToCheck & " — будний день.", vbInformation
   End If
   Exit Sub
End Sub
```

Процедуры событий также объявляют с помощью операторов `Sub...End Sub`. Они представляют специальный тип процедур, так как связаны со свойствами событий формы или отчета.

Подпроцедура `IsWeekday` получает аргумент `DateToCheck`. *Аргумент* — это значение, которое необходимо процедуре для обработки операторов. Подпроцедура `IsWeekday` отображает сообщение, которое указывает, является ли дата `DateToCheck` будним днем.

Если процедура не имеет аргументов, скобки оставляют пустыми. Например, объявленная ниже процедура не получает аргументов.

```
Public IsWeekday()
```

Понятие функции

Функции (*functions*) имеют много общего с подпроцедурами. Они состоят из ряда операторов, которые выполняют некоторое действие, и тоже могут получать аргументы. Однако, в отличие от подпроцедур, функции возвращают значение.

Функции начинаются с ключевых слов `Public Function` и заканчиваются оператором `End Function`. Если функция получает аргументы, их объявления указывают в скобках за ключевым словом `Function`. Первый оператор заканчивается объявлением значения, которое возвращает функция. Ниже приведен пример процедуры функции.

```
Public Function CalcTax(curPurchaseAmt As Currency, dblTaxRate As
Double) As Currency
   CalcTax = Round(curPurchaseAmt * dblTaxRate, 2)
Exit Function
End Function
```

Функция `CalcTax`, используемая для вычисления суммы налога, получает два аргумента: первый — сумма покупки, второй — текущая налоговая ставка. Ключевые слова `As Currency` в конце оператора объявления функции указывают на то, что функция возвращает денежное значение. Функция `CalcTax` вычисляет сумму налога от суммы покупки на основании заданной налоговой ставки. Функция возвращает вычисленное значение налога в процедуру, откуда была вызвана функция, присваивая результат названию функции (`CalcTax`).

Access предоставляет многочисленные встроенные процедуры для часто используемых операций, например, для работы с датами и временем, обработки строк и преобразования типов. Далее представлены некоторые примеры встроенных процедур.

- `Round()`: округляет число до указанного количества десятичных знаков.
- `IsWeekday()`: возвращает номер дня недели для указанной даты.
- `Now()`: возвращает текущие дату и время.
- `Left()`: возвращает указанное количество символов в строке, начиная с первой буквы.

Полный список всех доступных встроенных процедур см. в справочной системе Microsoft Visual Basic.

Осталось 20 минут

Понятие модуля

Перед созданием подпроцедуры или функции необходимо разобраться, что представляют собой модули. *Модуль* — это объект Access, в котором хранится коллекция процедур. Существует четыре типа модулей Access: модули форм, модули отчетов, стандартные модули и модули классов.

Модули форм и отчетов

Все процедуры событий для формы или отчета хранятся в *модуле формы* или *модуле отчета*. При создании первой процедуры события для формы или отчета Access автоматически создает модуль формы или отчета.

Чтобы просмотреть модуль формы или отчета, откройте форму или отчет в представлении Конструктор (Design). Затем выберите команду Вид⇨Программа (View⇨Code). Модуль будет отображен в редакторе Visual Basic (рис. 7.1).

Рис. 7.1. Просмотр модуля для формы `frm_MainMenu`

Модули формы и отчета предоставляют способ хранения в одном месте всего кода, который относится только к отдельной форме. Как правило, модули форм и отчетов содержат только процедуры событий, однако в них могут также храниться подпроцедуры и функции.

Стандартные модули

Стандартные модули позволяют хранить процедуры, которые следует использовать в любой форме или отчете приложения. Стандартные модули хранятся на вкладке Modules (Модули) в окне базы данных Access.

Чтобы просмотреть процедуры в модуле, выберите один из модулей, перечисленных на странице Модули (Modules), и щелкните на кнопке Конструктор (Design). Модуль отображается в редакторе Visual Basic (рис. 7.2).

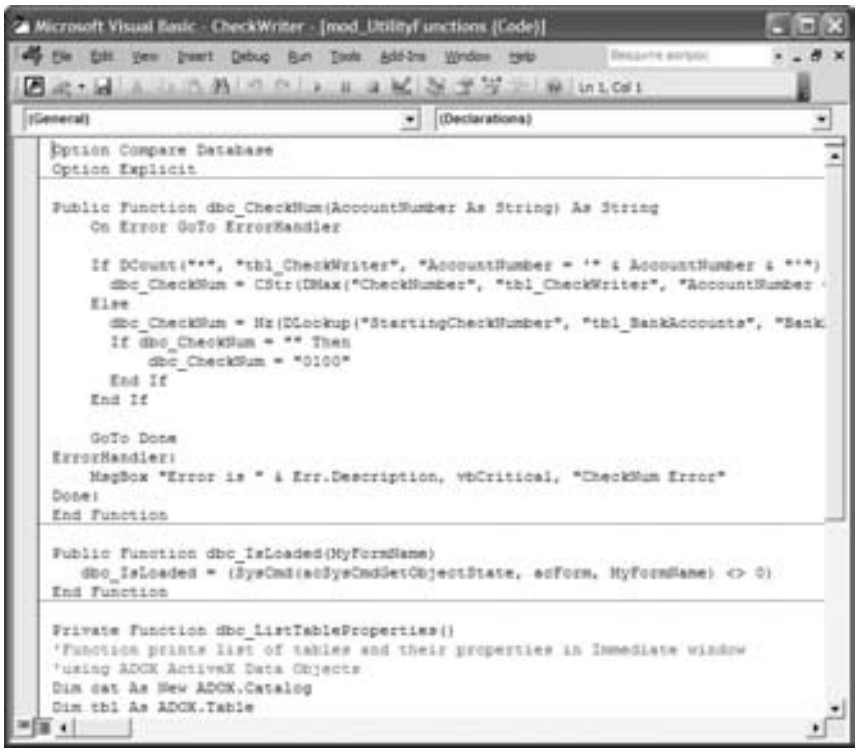

Рис. 7.2. Просмотр стандартного модуля `mod_UtilityFunctions` *в редакторе Visual Basic*

Модуль `mod_UtilityFunctions` содержит несколько общих процедур, которые можно использовать во всем приложении Check Writer. Эти процедуры включают следующие.

- `dbc_CheckNum()`: генерирует первый номер чека для формы Check Writer.
- `dbc_IsLoaded()`: определяет, открыта ли в данный момент указанная форма.

Несмотря на то, что функция `dbc_CheckNum()` относится к форме Check Writer, функция `dbc_IsLoaded()` может использоваться в любой форме, отчете или запросе в любом приложении.

Общие процедуры необходимо хранить в одном модуле, чтобы можно было легко импортировать модуль в другие приложения.

Модули классов

Модули классов (*class modules*) позволяют создавать новые специальные объекты для приложения. Для большинства приложений встроенные объекты интерфейса пользователя обеспечивают основу всех необходимых объектов. Однако для разработки более сложных приложений может понадобиться создание собственных объектов.

Модули классов хранятся на вкладке Модули окна баз данных Access вместе со стандартными модулями. Пиктограмма, которая отображается на странице Модули для модулей классов, отличается от пиктограммы стандартных модулей. На рис. 7.3 показана страница Модули, которая включает и стандартные модули, и модули классов. Модуль `cls_Setup` — это модуль класса, а остальные модули в списке — стандартные.

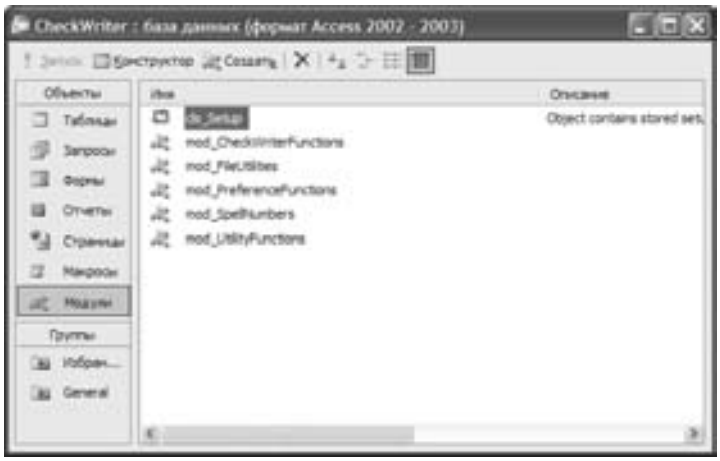

Рис. 7.3. На вкладке Модули для стандартных модулей и модулей классов отображаются разные пиктограммы

Чтобы просмотреть процедуры в модуле классов, выберите один из модулей, перечисленных на странице Модули, и щелкните на кнопке Конструктор. Модуль будет отображен в редакторе Visual Basic (рис. 7.4).

Модуль класса `cls_Setup` используется для работы с объектами начальных параметров приложения. Этот объект вы вправе использовать в любом приложении, в котором может понадобиться информация о компании и значения данных, отображаемые по умолчанию. Этот модуль класса применяется во всех приложениях, поскольку он не включает ссылки на какую-либо конкретную форму или отчет приложения.

Чтобы увидеть, как работает класс модуля `cls_Setup`, откройте форму `frm_CompanySetup` в базе данных Check Writer. В этой форме отображаются данные из таблицы `tbl_CompanySetup`. На рис. 7.5 показан код для события `On-Load` формы `frm_CompanySetup`.

Когда загружается форма `frm_CompanySetup`, функция `GetSetup` в модуле `cls_Setup` отображает значения для каждого элемента управления формы.

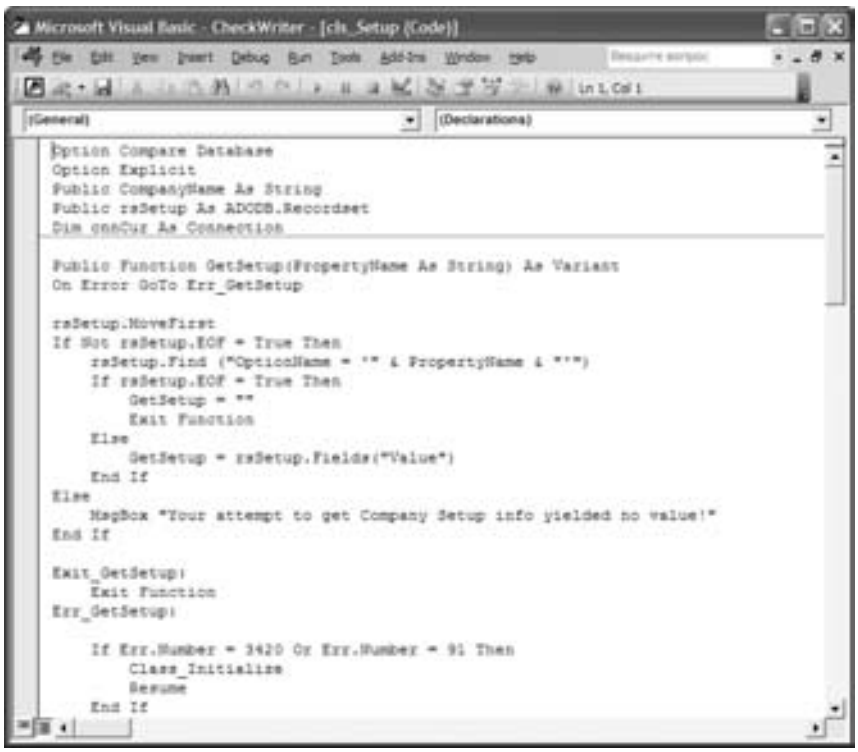

Рис. 7.4. Просмотр модуля класса `cls_Setup` *в редакторе Visual Basic*

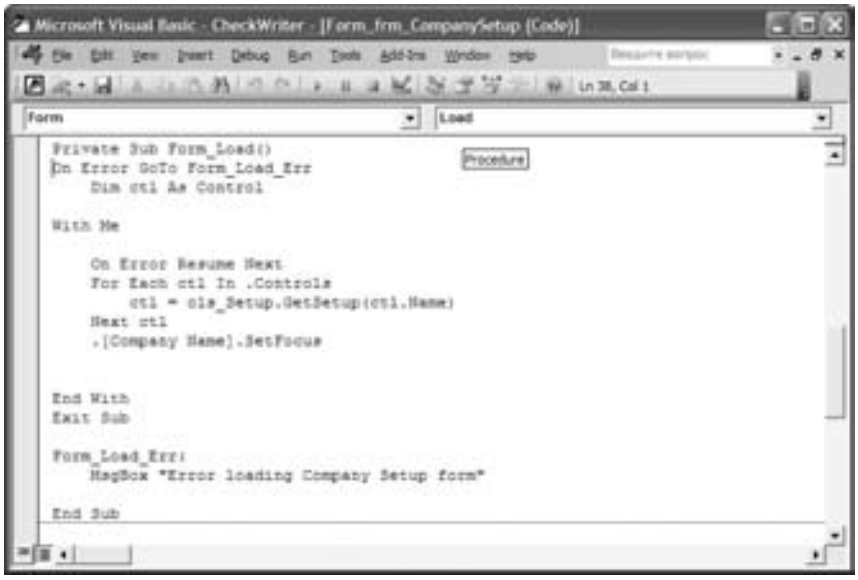

Рис. 7.5. Использование модуля класса для отображения начальных данных

Создание нового модуля

Чтобы создать новый стандартный модуль, выполните следующие действия.

1. Щелкните на кнопке объекта Модули в окне базы данных.
2. Щелкните на кнопке Создать (New) на панели инструментов.

Откроется редактор Visual Basic, после чего будет создан новый модуль под названием `Module1`. На рис. 7.6 показан новый модуль `Module1`.

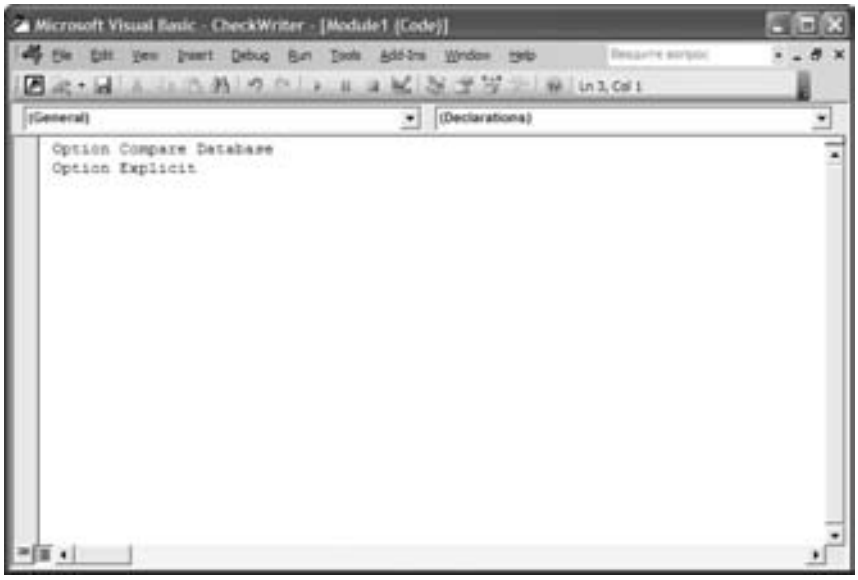

Рис. 7.6. Новый модуль в редакторе Visual Basic

При создании нового модуля в окне Code автоматически отображаются следующие два оператора: `Option Compare Database` и `Option Explicit`. Эти операторы необязательны, но их рекомендуется использовать во всех модулях. Оператор `Option Compare Database` указывает Access, какой порядок сортировки должен использоваться при сравнении данных в строках. Оператор `Option Explicit` делает обязательным объявление всех переменных во всех процедурах данного модуля.

При создании нового модуля автоматически создается раздел Declarations (Объявления). Оба оператора `Option` всегда помещаются в начало раздела объявлений. В этом разделе могут содержаться переменные, которые должны быть доступны во всех процедурах текущего модуля или в любом модуле приложения.

Осталось 10 минут

Создание новой процедуры

После заполнения раздела объявлений можно переходить к созданию процедуры. Выполните следующие действия, чтобы создать процедуру под названием `ShowMessage`.

Рис. 7.7. Использование полей со списками для выбора раздела модуля

1. Выберите команду Insert⇨Procedure в меню редактора Visual Basic. Появится диалоговое окно Add Procedure (рис. 7.8).

Рис. 7.8. Создание новой процедуры

2. Введите **ShowMessage** в поле названия новой процедуры. Выберите параметр Sub в разделе Type и параметр Option в разделе Scope.
3. Щелкните на кнопке OK. Новая процедура будет отображена в окне Code.

Новая процедура должна выглядеть приблизительно так, как на рис. 7.9.

Операторы процедуры вводятся между операторами Sub и End Sub. Введите в теле процедуры ShowMessage следующие операторы.

```
Dim MsgTxt As String
MsgTxt = "Выполнение процедуры ShowMessage."
Beep
MsgBox MsgTxt, vbInformation, "ShowMessage Procedure"
```

Рис. 7.9. Новая процедура в окне Code

Готовая процедура должна иметь вид как на рис. 7.10.

Рис. 7.10. Процедура ShowMessage

Чтобы увидеть, как работает процедура ShowMessage, запустите ее из редактора Visual Basic. Откройте окно Immediate, введите в нем **ShowMessage** и нажмите <Enter>. При выполнении процедуры ShowMessage прозвучит звуковой сигнал и появится сообщение.

Перед выполнением процедур их необходимо всегда компилировать.

Использование процедур в форме

Созданную процедуру можно использовать в любой форме, отчете или запросе. Чтобы применить процедуру в форме или отчете, необходимо просто ввести название процедуры как оператор или часть оператора в одной из процедур событий. При вводе названия процедуры следует обязательно указать аргументы процедуры в скобках после имени процедуры.

Событие Открытие (OnOpen) для формы Check Writer вызывает функцию dbc_IsLoaded, которая хранится в модуле mod_UtilityFunctions. Следующий код отображает раздел программы для события Открытие формы Check Writer.

```
If dbc_IsLoaded("frm_CheckReconciliation") Then
   Me![BankAccount] =
[Forms]![frm_CheckReconciliation]![BankAccount]
Else
   Me![BankAccount] = DLookUp("[DefaultBankAccountNumber]",
"[tbl_Preferences]")
End If
```

Когда процедура события Открытие вызывает функцию dbc_IsLoaded, она передает в виде аргумента название формы для проверки — frm_CheckReconciliation. При выполнении функция dbc_IsLoaded проверяет, открыта ли форма frm_CheckReconciliation. Если форма открыта, функция возвращает True; в противном случае — False. На рис. 7.11 показана программа для функции dbc_IsLoaded.

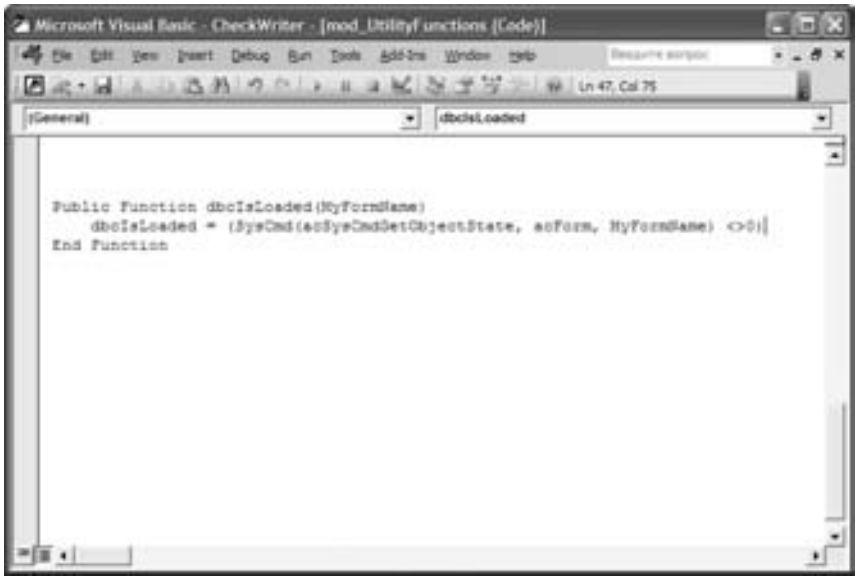

Рис. 7.11. Получение параметра в процедуре

Функция dbc_IsLoaded получает параметр MyFormName. Этот параметр используется для указания названия объекта, который необходимо проверить — в данном случае это форма frm_CheckReconciliation. Функция dbc_IsLoaded вызывает встроенную функцию SysCmd для проверки состояния объекта, указанного в MyFormName.

Использование процедур в запросах

Функции могут применяться в запросах, поскольку они возвращают значение. Вы можете включать процедуры функций в запросы, чтобы преобразовать данные или вычислить выражения, а также использовать в запросе встроенную функцию (`Left`, `Date`, `Now`) либо включить в него одну из собственных функций. Чтобы применить функцию в запросе, необходимо просто передать название столбца как аргумент функции. На рис. 7.12 показан запрос, который вызывает функцию `Date`.

Рис. 7.12. Вызов функции из запроса

Обзор

Данный урок был посвящен работе с процедурами и модулями. Мы рассмотрели такие темы.

- Существует два типа процедур: подпрограммы и функции.
- Функции возвращают значения. И подпрограммы, и функции могут получать аргументы.
- Процедуры хранятся в модулях.
- Модули классов используются для создания определений специальных объектов. Все остальные процедуры хранятся в стандартных модулях.
- Вы можете вызывать функции и подпроцедуры из форм и отчетов. В запросе позволяется вызывать только процедуры функций.

Проверьте себя

1. Назовите компонент подпроцедуры, который указывает, что процедура должна получать значение из другой программы. (См. раздел “Понятие подпрограммы”.)
2. Чем отличаются подпрограммы и функции? (См. раздел “Понятие функции”.)
3. Назовите объект Access, который используется для хранения процедур. (См. раздел “Понятие модуля”.)
4. Перечислите четыре типа модулей. (См. раздел “Понятие модуля”.)
5. Назовите две причины, по которым в процедурах следует использовать операторы `Option Explicit`. (См. раздел “Создание нового модуля”.)