А. Гарнаев

САМОУЧИТЕЛЬ

VBA

Технология создания пользовательских приложений

- *От моделирования до автоматизации повседневных работ*
- *Основные элементы, объекты и методы Visual Basic for Applications*
- *Примеры разработки приложений*
- *Применение VBA при работе с базами данных*

bhv
www.bhv.ru

Андрей Гарнаев

САМОУЧИТЕЛЬ VBA

Дюссельдорф Киев
Москва Санкт-Петербург

УДК 681.3.06

Настоящая книга является, с одной стороны, подробным справочником по Visual Basic for Applications (VBA), а с другой стороны, самоучителем по составлению и разработке приложений с помощью этого языка. Большое количество примеров позволит быстро овладеть практическими приемами программирования и эффективно решать разнообразные задачи. В книге приводится подробное описание средств и возможностей VBA. В каждом из предлагаемых уроков разрабатывается пример пользовательского приложения и дается подробный анализ всех программ, которые, кроме того, снабжены поясняющими комментариями. В конце каждого урока имеется самостоятельное задание для закрепления изученного материала.

Для широкого круга программистов и пользователей

Группа подготовки издания:

Главный редактор	*Екатерина Кондукова*
Ответственный редактор	*Наталья Таркова*
Компьютерная верстка	*Наталья Смирнова*
Корректура	*Светлана Журавина*
Дизайн обложки	*Наталья Смирнова*
Производство	*Николай Тверских*

Гарнаев А. Ю.
Самоучитель VBA. — СПб.: БХВ — Санкт-Петербург, 1999. — 512 с., ил.

ISBN 5-8206-0067-3

Лицензия ЛР № 065953 от 15.06.98. Подписано в печать 09.07.99.
Формат 70×100 $^{1}/_{16}$. Печать офсетная. Усл. печ. л. 41,28.
Тираж 3000 экз. Заказ
БХВ — Санкт-Петербург, 198005, С.-Петербург, Измайловский пр., 29.

Отпечатано с фотоформ
в типографии «Наука» РАН
199034, Санкт-Петербург, 9 линия, 12.

Содержание

ПРЕДИСЛОВИЕ 1
Краткий обзор материала книги 2
ВВЕДЕНИЕ 5
Зачем нужен VBA 5
Создание функций пользователя 10

ЧАСТЬ I. ОСНОВНЫЕ СРЕДСТВА И ВОЗМОЖНОСТИ VBA 13
ГЛАВА 1. ОСНОВНЫЕ ЭЛЕМЕНТЫ VBA 15
Что такое VBA 15
Объекты и их семейства 15
Объекты OLE и ActiveX 16
Классы 16
Иерархия объектов 17
Методы 18
Свойства 18
События 19
Структура редактора VBA 20
Окно проекта 20
Окно для редактирования кода 21
Окно редактирования форм (UserForm) 25
Окно свойств 27
Окно Просмотр объектов (Object Browser) 28
ГЛАВА 2. ОСНОВНЫЕ ОБЪЕКТЫ VBA 30
Объект Application 30
Свойства объекта Application 30
Методы объекта Application 33
События объекта Application 35
Объект Workbook и семейство Workbooks 36
Свойства объекта Workbook и семейства Workbooks 36
Методы объекта Workbook и семейства Workbooks 37
События объекта Workbook и семейства Workbooks 39
Объект Worksheet и семейство Worksheets 39
Свойства объекта Worksheet и семейства Worksheets 39
Методы объекта Worksheet и семейства Worksheets 40
События объекта Worksheet 44
Объекты Range и Selection 44
Адресация ячеек 44
Задание групп строк и столбцов с помощью объекта Range 45
Связь объекта Range и свойства Cells 45
Свойства и методы объекта Range 46

ГЛАВА 3. МЕТОДЫ ОБЪЕКТА RANGE, ИСПОЛЬЗУЮЩИЕ КОМАНДЫ EXCEL 55

Метод DataSeries 55
Метод AutoFill 57
Метод AutoFilter 59
Метод AdvancedFilter 62
Метод Consolidate 65
Метод Find 68
Метод GoalSeek 69
Метод Sort 72
Метод Subtotal 75

ГЛАВА 4. СЦЕНАРИИ И ОПРЕДЕЛЕНИЕ СТРУКТУРЫ ДАННЫХ 82

Объект Scenario 82
Объект Outline 86

ГЛАВА 5. ДИАГРАММЫ 91

Объекты Chart и ChartObject 91
Свойства объекта Chart 92
Методы объекта Chart 94
Линия тренда 103

ГЛАВА 6. СВОДНЫЕ ТАБЛИЦЫ 108

Объект PivotTable 108
Метод PivotTableWizard 109
Преобразование сводной таблицы 117

ГЛАВА 7. ЭЛЕМЕНТЫ УПРАВЛЕНИЯ И ПОЛЬЗОВАТЕЛЬСКАЯ ФОРМА 121

Элементы управления 121
Режим конструктора 122
Установка свойств элемента управления 123
Редактор кода 124
Пользовательская форма UserForm 126
Семейство Controls 127
Создание пользовательской формы 127
Общие свойства элементов управления 130
Соглашения об именах 132
Общие методы и события элементов управления 133
Объект DataObject 136
Поле 137
Надпись 139
Кнопка 140
Список 141
Заполнение списка 143
Выбор нескольких элементов из списка 145
Поле со списком 146
Полоса прокрутки и счетчик 147
Переключатель 148

Рамка ... 150
Флажок и выключатель ... 150
Рисунок ... 152
Ссылки на ячейки и диапазоны ... 155
Набор страниц ... 155
Набор вкладок ... 156
Дополнительные элементы управления ... 157
Последовательность выбора элементов управления ... 160
Инициализация и отображение диалогового окна ... 160
Закрытие диалогового окна ... 163
Отображение встроенных диалоговых окон ... 163

ГЛАВА 8. ПРОГРАММИРОВАНИЕ ПАНЕЛИ ИНСТРУМЕНТОВ ... 165

Объект CommandBar и семейство CommandBars ... 165
Семейство CommandBarControls и объект CommandBarControl ... 168
Пример создания панели инструментов пользователя ... 170
Пример создания строки меню пользователя ... 173
Создание пользовательской панели инструментов вручную ... 175
Удаление элемента управления из панели инструментов вручную ... 178
Удаление пользовательской панели инструментов вручную ... 178
Назначение вручную макроса кнопке ... 179
Изменение и создание вручную изображения на кнопке ... 180

ГЛАВА 9. ПРОГРАММИРОВАНИЕ СРЕДСТВ ДЛЯ РАБОТЫ СО СПРАВОЧНОЙ ИНФОРМАЦИЕЙ ... 183

Структура помощника ... 183
Типы помощника ... 184
Свойства объекта Assistant ... 185
Объект Balloon ... 186

ГЛАВА 10. РАБОТА С ГРАФИЧЕСКИМИ ОБЪЕКТАМИ ... 191

Семейства Shapes и ShapeRange ... 191
Методы, создающие объекты Shape ... 192

ГЛАВА 11. ОСНОВЫ ПРОГРАММИРОВАНИЯ НА VBA ... 199

Типы данных ... 199
Описание переменных ... 200
Допустимые имена ... 202
Инструкция DefТип ... 202
Массивы ... 203
Динамические массивы ... 204
Функции и процедуры для работы с массивами ... 205
Константы ... 206
Тип данных, определенный пользователем ... 207
Операции VBA ... 209
Приоритеты операций ... 210
Встроенные функции VBA ... 211
Математические функции ... 211
Функции проверки типов ... 212
Функции преобразования форматов ... 213

Функции обработки строк ... 214
Функции времени и даты ... 217
Функции выбора ... 220
Функции, возвращающие строки ... 221
Встроенные диалоговые окна ... 221
Инструкции VBA ... 227
Оператор присвоения ... 227
Перенос строки ... 228
Комментарии ... 228
Расположение нескольких операторов на одной строке ... 228
Операторы перехода и выбора ... 229
Операторы повтора ... 230
Условная компиляция ... 232
Процедура ... 233
Переход в подпрограмму и возвращение из подпрограммы ... 236
Вызов процедуры ... 236
Назначение значений по умолчанию необязательным параметрам ... 239
Использование неопределенного количества параметров ... 239
Рекурсивные процедуры ... 239
Область определения переменной ... 240
Время жизни переменной ... 241

ГЛАВА 12. ПРОЦЕДУРЫ ОБРАБОТКИ ОШИБОК И ОТЛАДКА ПРОГРАММ ... 242

Разработка процедур, предотвращающих появление ошибок ... 242
Перехват и обработка ошибок ... 246
Отладка программ ... 251
Ошибки компиляции ... 251
Ошибки выполнения ... 253
Логические ошибки ... 254
Инструкция Option Explicit ... 255
Пошаговое выполнение программ ... 255
Точка останова ... 256
Вывод значений свойств и переменных ... 256

ГЛАВА 13. РАБОТА С ФАЙЛАМИ ... 259

Типы файлов в VBA ... 259
Открытие и закрытие файла ... 259
Ввод данных в файл последовательного доступа ... 261
Вывод данных из файла последовательного доступа ... 263
Работа с файлом произвольного доступа ... 265
Наиболее употребляемые инструкции и функции при работе с файлами ... 268
Объект FileSearch ... 270

ГЛАВА 14. ПОЛЬЗОВАТЕЛЬСКИЕ ОБЪЕКТЫ ... 273

Создание модулей класса ... 273
Процедуры Property Let, Property Set и Property Get ... 274
Пример создания класса ... 275

ГЛАВА 15. РАБОТА С ВНЕШНИМИ БАЗАМИ ДАННЫХ 280

Создание запросов с помощью Microsoft Query 280
Что такое ODBC? 286
Microsoft Jet 287
DAO — объектный доступ к данным 287
Порядок работы при объектном доступе к данным 288
Создание рабочей области 288
Открытие базы (источника) данных 289
Объект Recordset 290
Пример приложения 295

ЧАСТЬ II. ПРАКТИЧЕСКИЕ ПРИЕМЫ ПРОГРАММИРОВАНИЯ НА VBA 303

УРОК 1. ТЕМА: ИГРА "ОРЕЛ И РЕШКА" 305

Цель урока 305
Теория 305
Практика 306
Самостоятельное задание 312

УРОК 2. ТЕМА: РАСЧЕТ МАРГИНАЛЬНОЙ ПРОЦЕНТНОЙ СТАВКИ 314

Цель урока 314
Теория 314
Практика 316
Самостоятельное задание 325

УРОК 3. ТЕМА: РАБОТА СО СПИСКОМ 327

Цель урока 327
Практика 327
Самостоятельное задание 331

УРОК 4. ТЕМА: РАСЧЕТ АМОРТИЗАЦИИ 333

Цель урока 333
Теория 333
Практика 336
Самостоятельное задание 346

УРОК 5. ТЕМА: РЕШЕНИЕ УРАВНЕНИЯ, ЗАВИСЯЩЕГО ОТ ПАРАМЕТРА. ПОСТРОЕНИЕ ДИАГРАММЫ 348

Цель урока 348
Теория 348
Практика 349
Самостоятельное задание 360

УРОК 6. ТЕМА: УПРАВЛЕНИЕ РАЗМЕРОМ И ПЕРЕМЕЩЕНИЕМ ЭЛЕМЕНТОВ УПРАВЛЕНИЯ 361

Цель урока 361
Теория 361
Практика 362
Пример управления размером и перемещением элемента управления 368

Перемещение элемента управления при помощи операции drag-and-drop ... 370
Самостоятельное задание ... 373
УРОК 7. ТЕМА: ЗАПОЛНЕНИЕ БАЗЫ ДАННЫХ ... 374
Цель урока ... 374
Практика ... 374
Самостоятельное задание ... 386
УРОК 8. ТЕМА: ПОСТРОЕНИЕ ПОВЕРХНОСТИ ... 388
Цель урока ... 388
Практика ... 388
Самостоятельное задание ... 402
УРОК 9. ТЕМА: ПЕРИОДИЧЕСКИЕ ВЫПЛАТЫ. ПОСТРОЕНИЕ ДИАГРАММ ... 404
Цель урока ... 404
Теория ... 404
Практика ... 405
Самостоятельное задание ... 418
УРОК 10. ТЕМА: ЕЩЕ РАЗ О СОСТАВЛЕНИИ БАЗЫ ДАННЫХ ... 419
Цель урока ... 419
Практика ... 419
Самостоятельное задание ... 446
УРОК 11. ТЕМА: ИГРА В КРЕСТИКИ И НОЛИКИ ... 450
Цель урока ... 450
Практика ... 450
Самостоятельное задание ... 464
УРОК 12. ТЕМА: ЛИНИЯ ТРЕНДА ... 466
Цель урока ... 466
Теория ... 466
Практика ... 468
Самостоятельное задание ... 480
УРОК 13. ТЕМА: СОСТАВЛЕНИЕ РАСПИСАНИЯ ... 481
Цель урока ... 481
Практика ... 481
Самостоятельное задание ... 488
УРОК 14. ТЕМА: РАБОТА С ФАЙЛАМИ ... 489
Цель урока ... 489
Практика ... 489
Файл последовательного доступа ... 489
Файл произвольного доступа ... 493
Создание заставки приложения ... 501

Предисловие

Настоящая книга является с одной стороны, подробным справочником по Visual Basic for Applications (VBA), а с другой стороны, самоучителем по составлению и разработке приложений, написанных на этом языке. Это уникальное сочетание, которое, следуя рекламному подходу, можно назвать "два в одном", обеспечивает большую гибкость при решении читателем своих собственных задач. Самоучитель на большом количестве примеров умело и доступно обучает, как можно быстро и эффективно решать разнообразные задачи. В справочнике приводится подробное описание возможностей VBA, имея такие сведения под рукой у читателя исчезнет необходимость бегать по магазинам в поиске дополнительной литературы при написании самостоятельных приложений, что несомненно сбережет время и кошелек.

Самоучитель состоит из уроков. В каждом из уроков разрабатывается пример пользовательского приложения и дается подробный анализ. Тексты всех программ снабжены доскональными комментариями. Можно сказать, что все рассматриваемые программы разложены буквально по маленьким разжеванным кусочкам, которые читателю только и остается проглотить. По завершению урока предлагается самостоятельное задание, выполнение которого поможет лучше закрепить разобранный материал.

С помощью VBA можно легко и быстро создавать пользовательские приложения, используя единую для всех офисных программ среду и язык. Научившись разрабатывать приложения для одной офисной программы, например Excel (которой, как наиболее популярной офисной программе, в основном и посвящена данная книга), можно создавать приложения и для других офисных программ, например Access. Внимательно читая эту книгу, можно стать искусным разработчиком и научиться пользоваться мощными средствами разработки приложений Excel для того, чтобы конструировать эффективные и применимые к реальной жизни приложения. Кроме того, по своей структуре, интерфейсу и синтаксису VBA образует ядро Visual Basic. Поэтому тот, кто изучит программирование на VBA очень быстро может освоить и Visual Basic.

В данной книге уделяется огромное внимание программированию на языке VBA, но это совсем не требует от читателя быть профессиональным программистом. VBA обладает мощными встроенными интеллектуальными средствами, которые позволяют даже начинающему пользователю быстро самостоятельно разрабатывать профессиональные приложения. Например, при написании кода программы редактор VBA сам предлагает пользователю возможные продолжения составляемых им инструкций. Другим примером встроенных интеллектуальных средств VBA является макрорекордер, кото-

рый переводит все выполняемые вручную пользователем действия в основном приложении на язык VBA. Таким образом, макрорекордер позволяет пользователю поручать VBA самому создавать большие куски кода разрабатываемого приложения.

Краткий обзор материала книги

Книга состоит из двух частей, первая из которых имеет 15 глав со следующим кратким содержанием:

- Во введении на простейших примерах объясняется, зачем нужен VBA.
- Глава 1 отвечает на вопрос: "Что такое VBA?". В ней также дано описание основных структурных элементов VBA.
- В главе 2 рассматриваются основные элементы объектной иерархической структуры VBA.
- В главах 3 и 4 дан обзор методов VBA, программирующих команды для работы с рабочим листом, которые позволяют строить прогрессии, создавать фильтры и консолидировать данные, организовывать сценарии и структуры, решать уравнения, подводить промежуточные итоги и сортировать данные.
- Глава 5 описывает, как строятся диаграммы в VBA.
- В главе 6 обсуждается, как программировать одно из наиболее мощных средств по анализу данных — сводные таблицы.
- В глава 7 приведен обзор элементов управления VBA и описание того, как в VBA создаются и программируются диалоговые окна.
- Глава 8 продолжает начатый в предыдущей главе разговор по созданию пользовательского интерфейса. В ней объясняется, как создать пользовательское меню и панели инструментов.
- Глава 9 посвящена вопросу создания очень полезного и наглядного средства Microsoft Office — помощника.
- В главе 10 дано обзорное описание процесса создания графических объектов.
- В главе 11 приводится краткая информация по основным понятиям языка VBA: какими типами данных оперирует VBA, что такое переменная, константа, массив и динамический массив, как создается пользовательский тип переменной. В ней перечислены операции, встроенные функции, операторы и процедуры VBA, а также типы процедур.
- В главе 12 обсуждаются принципы создания процедур обработки ошибок, а также встроенные в редактор VBA мощные средства по отладке программ.

- В главе 13 описаны типы файлов и способы работы с ними в VBA.
- Глава 14 объясняет, как в VBA можно создавать пользовательские объекты.
- В главе 15 дан обзор методов по работе с внешними базами данных, использовании Microsoft Query, открытой связи с базой данных (ODBC) и объектов доступа к данным (DAO).

Во второй части приведено 14 уроков самоучителя по созданию пользовательских приложений:

- В уроке 1 на примере создания приложение по игре в орел и решку показывается: как в редакторе VBA создается программа и как она запускается на выполнение; создание пользовательского диалогового окна; программный контроль за вводом в поле чисел, а не строковой информации; программное управление запретом ввода данных в поле; работа с функцией генератора случайных чисел; вывод числовой информации в поле.
- В уроке 2 на примере разработки приложения по расчету маргинальной процентной ставки объясняется: как программно решаются уравнения; программный ввод формулы в ячейку рабочего листа; финансовые функции; проверка корректности ввода данных из диалогового окна; назначение клавишам <Enter> и <Esc> функций кнопок диалогового окна; создание всплывающих подсказок у элементов управления; использование MacroRecorder для упрощения и убыстрения написания кода; программное форматирование ячеек рабочего листа.
- В уроке 3 на рассмотренном примере работы со списком показывается: как заполняется список; управление выбором нескольких элементов из списка; как выполнить специфицированную операцию над выбранными элементами из списка с помощью переключателей.
- В уроке 4 на примере разработки приложения по расчету амортизации объясняются: финансовые функции расчета амортизации; управление видимостью отдельных элементов управления в окне диалога; программный вывод объектов WordArt на рабочий лист.
- В уроке 5 на рассмотренном примере показывается: ввод формул при помощи элемента управления `RefEdit`; нахождение корня уравнения зависящего от параметра; установка параметров метода `GoalSeek`; создание прогрессий на рабочем листе; программирование протаскивания маркера заполнения выделенного диапазона на рабочем листе; построение диаграмм.
- В уроке 6 на обсужденных примерах показывается: программное управление размерами диалогового окна и элементов управления; задание последовательности элементов управления в виде массива объектов; определение текущего объема вклада; задание параметров счетчика; как можно программно или при помощи drag-and-drop операции перемещать элементы управления по поверхности диалогового окна.

- В уроке 7 на примере конструируемого приложения демонстрируется: как при помощи диалогового окна можно заполнить базу данных на рабочем листе; программирование примечаний и текстовых полей на рабочем листе; использование переключателя и флажков; создание пользовательского заголовка окна приложения и программное закрепление области.
- В уроке 8 на примере разработки приложения по построению поверхности объясняется: как табулируются функции, зависящие от двух аргументов; преобразование формулы с аргументами *x* и *y* в формулу рабочего листа; программное построение поверхности; запись диаграммы в графический файл; считывание графического файла в элемент управления `Image`; программное управление углом зрения, под которым смотрят на поверхность, и углом поворота поверхности вокруг оси Z.
- В уроке 9 на примере конструируемого приложения по расчету периодических выплат показывается: как используется финансовая функция `ПЛАТ (PMT)`; вывод результатов табулирования функции в элемент управления `ListBox` (список); построение диаграммы, тип которой выбирается в группе переключателей; программная проверка наличия файла на диске.
- В уроке 10 на примере разработки приложения по работе с базой данных демонстрируется: конструирование пользовательского интерфейса; создание приложения, работающего с несколькими диалоговыми окнами; поиск информации в базе данных; редактирование записей в базе данных; удаление ненужных записей из базы данных; архивация данных; программирование фильтрации и сортировки данных; создание сводных таблиц; добавление пользователем новых элементов в список с полем во время выполнения программы.
- В уроке 11 на примере игры в крестики и нолики объясняется: удаление рисунка из элемента управления; учет количества щелчков по элементу управления; управление видимостью границы элемента управления; создание игрового поля.
- В уроке 12 на примере приложения по построению линии тренда показывается: конструирование многостраничных диалоговых окон и линии тренда; применение метода `Offset` для вывода данных на рабочем листе; считывание данных из каждой отдельной ячейки диапазона.
- В уроке 13 на примере приложения по составлению расписания обсуждается: передача информации между элементами управления при обработке события `Click`; управление видимостью рисунков и цветом элементов управления.
- В уроке 14 на примере показана работа с текстовыми файлами: считывание и запись в файл последовательного доступа; считывание и запись записей в файла прямого доступа; создание и работа с пользовательскими типами данных; создание простейшего текстового редактора и заставки приложения.

Введение

Зачем нужен VBA

В качестве первоначального знакомства с VBA попытаемся решить следующую задачу. Допустим, вы решили вести учет своих расходов, и с этой целью в конце каждого месяца намерены составлять таблицу (рис. В.1) и строить диаграмму для более наглядного отображения доли каждой статьи расходов вашего бюджета. Составлять ежемесячно одну и ту же таблицу с одновременным построением диаграммы довольно непроизводительная трата времени. Более разумно один раз научить компьютер создавать таблицу, а потом по мере необходимости лишь отдавать команду подготовки таблицы, чтобы осталось только внести в нее данные.

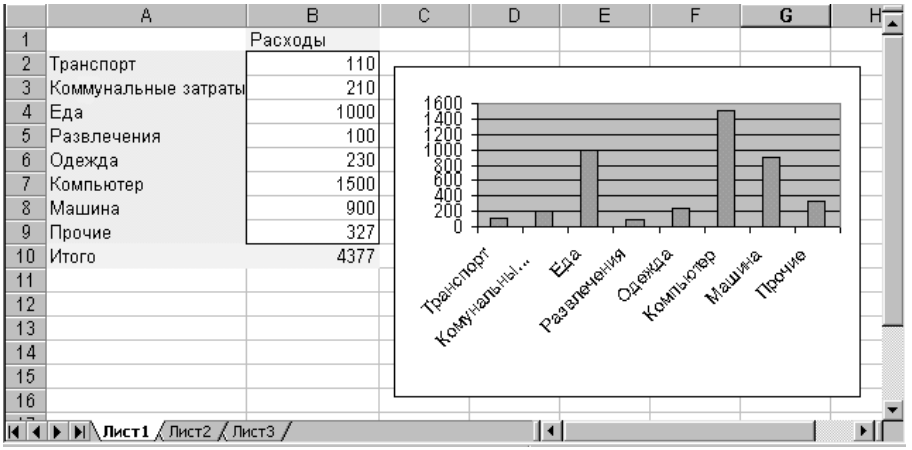


Рис. В.1. Таблица ежемесячных расходов

Для обучения компьютера отлично подходит MacroRecorder — транслятор, создающий программу (макрос) на языке VBA, которая является результатом перевода на язык VBA действий пользователя с момента запуска MacroRecorder до окончания записи макроса.

Итак, для активизации MacroRecorder выберите команду **Сервис**, **Макрос**, **Начать запись** (Tools, Macro, Record New Macro). Появится диалоговое окно **Запись макроса** (Record Macro) (рис. В.2). Это диалоговое окно позволяет задать параметры макроса.

Поля **Имя макроса** (Macro Name) и **Описание** (Description) предназначены для задания имени макроса и его описания. Последнее важно для многократно используемых макросов. Наша память не долговечна и, не имея под-

сказки в виде описания, через некоторое время бывает трудно вспомнить, для чего тот или иной макрос создавался. По умолчанию макросам присваиваются имена `Макрос1`, `Макрос2` и т. д. Чтобы было легче разпознать макрос, лучше не оставлять стандартное, а присвоить ему какое-нибудь уникальное имя, поясняющее его назначение. В данном случае присвоим макросу, например, имя `Расходы`. Поле **Сочетание клавиш** (Shortcut Key) позволяет назначить макросу комбинацию клавиш, т. е. указать клавишу, которая в сочетании с клавишей <Ctrl> будет служить для запуска его на выполнение. Назначать комбинацию клавиш макросу совсем не обязательно. Это стоит делать только для постоянно используемых макросов, для быстрого доступа к ним. Макрос всегда можно вызвать командой **Сервис**, **Макрос** (Tools, Macro). Раскрывающийся список **Сохранить в книге** (Store Macro in) предназначен для выбора книги, в которой будет сохранен макрос. Если выбрать **Личная книга макросов** (Personal Macro Workbook), то макрос будет сохранен в специальной скрытой книге. Эта книга хотя и скрыта, но является открытой, и записанные в ней макросы доступны для других рабочих книг. Команда **Окно**, **Отобразить** (Window, Unhide) позволяет отобразить личную книгу макросов. Если в раскрывающемся списке выбрать **Эта книга** (This Workbook) (этот выбор по умолчанию предлагает компьютер), то макрос сохранится на новом листе модуля в активной рабочей книге, а если выбрать **Новая книга** (New Workbook) — в новой рабочей книге.

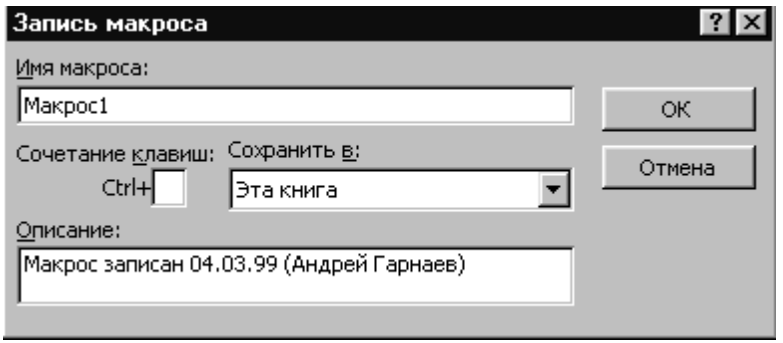


Рис. В.2. Диалоговое окно **Запись макроса**

Итак, в диалоговом окне **Запись макроса** (Record Macro) в поле **Имя макроса** (Macro Name) введем `Расходы`, а в поле **Описание** (Description) — `Расчет месячных расходов` и нажмем кнопку **ОК**. Появится плавающая панель инструментов с кнопкой ■ **Остановить запись** (Stop Recording). Теперь все производимые действия будут записываться до тех пор, пока не будет нажата эта кнопка. Построим шаблон таблицы расходов по следующему алгоритму:

1. Активизируем ячейку `B1` и введем в нее `Расходы`.
2. Активизируем ячейку `A2` и введем в нее `Транспорт`.

3. Активизируем ячейку `A3` и введем в нее `Коммунальные`.
4. Активизируем ячейку `A4` и введем в нее `Еда`.
5. Активизируем ячейку `A5` и введем в нее `Развлечения`.
6. Активизируем ячейку `A6` и введем в нее `Одежда`.
7. Активизируем ячейку `A7` и введем в нее `Компьютер`.
8. Активизируем ячейку `A8` и введем в нее `Машина`.
9. Активизируем ячейку `A9` и введем в нее `Прочие`.
10. Активизируем ячейку `A10` и введем в нее `Итого`.
11. Активизируем ячейку `B10` и введем в нее формулу `=СУММ(B2:B9)`, вычисляющую суммарные расходы.
12. Выберем диапазон `B2:B9` и при помощи раскрывающегося списка **Границы** (Borders) панели инструментов **Форматирование** (Formatting) создадим рамку, окаймляющую этот диапазон.
13. Выберем диапазон `A10:B10` и при помощи раскрывающегося списка **Цвет заливки** (Fill Color) панели инструментов **Форматирование** (Formatting) окрасим этот диапазон в желтый цвет.
14. Выберем ячейку `B1` и при помощи раскрывающегося списка **Цвет заливки** (Fill Color) панели инструментов **Форматирование** (Formatting) окрасим эту ячейку в желтый цвет.
15. Выберем диапазон `A2:A9` и при помощи раскрывающегося списка **Цвет заливки** (Fill Color) панели инструментов **Форматирование** (Formatting) окрасим этот диапазон в светло-бирюзовый цвет.
16. Выберем столбец `A`, изменим его ширину так, чтобы введенный в диапазон `A2:A9` текст помещался в этом столбце.
17. Выберем диапазон `A2:B9` и при помощи мастера диаграмм, вызываемого кнопкой **Мастер диаграмм** (Chart Wizard) панели инструментов **Стандартная** (Standard), создадим диаграмму.

Рабочий лист теперь будет выглядеть так, как показано на рис. В.1. Остановим запись макроса, нажав кнопку  **Остановить запись** (Stop Recording). Заполним ячейки таблицы исходными данными, расчет суммарных расходов и построение диаграммы теперь будет происходить автоматически.

Для просмотра записанной процедуры необходимо выбрать команду **Сервис**, **Макрос**, **Макросы** (Tools, Macro, Macros), которая вызовет диалоговое окно **Макрос** (Macro) (рис. В.3).

В этом диалоговом окне в списке выделим макрос и нажмем кнопку **Изменить** (Edit). Это вызовет появление главного окна редактора VBA (рис. В.4). Ниже приведен полный текст записанного макроса.

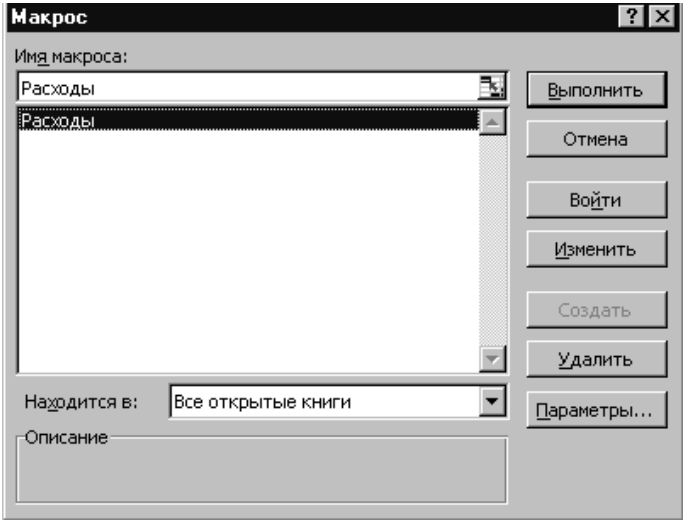


Рис. В.3. Диалоговое окно **Макрос**

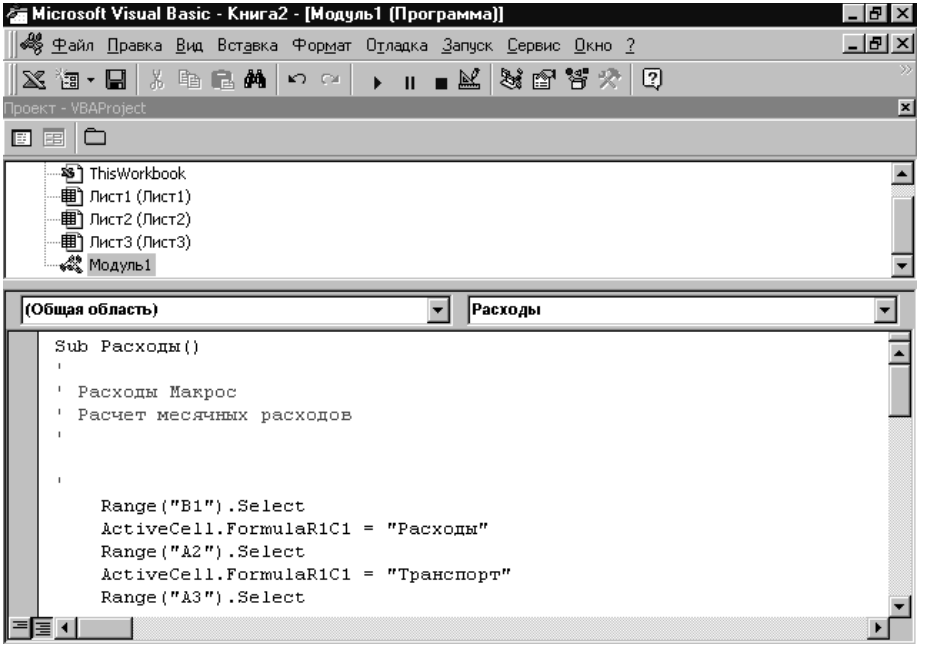


Рис. В.4. Главное окно редактора VBA

```
Sub Расходы()
'
' Расходы Макрос
```

```
' Расчет месячных расходов
'

'
    Range("B1").Select
    ActiveCell.FormulaR1C1 = "Расходы"
    Range("A2").Select
    ActiveCell.FormulaR1C1 = "Транспорт"
    Range("A3").Select
    ActiveCell.FormulaR1C1 = "Коммунальные"
    Range("A4").Select
    ActiveCell.FormulaR1C1 = "Еда"
    Range("A5").Select
    ActiveCell.FormulaR1C1 = "Развлечения"
    Range("A6").Select
    ActiveCell.FormulaR1C1 = "Одежда"
    Range("A7").Select
    ActiveCell.FormulaR1C1 = "Компьютер"
    Range("A8").Select
    ActiveCell.FormulaR1C1 = "Машина"
    Range("A9").Select
    ActiveCell.FormulaR1C1 = "Прочие"
    Range("A10").Select
    ActiveCell.FormulaR1C1 = "Итого"
    Range("B10").Select
    ActiveCell.FormulaR1C1 = "=SUM(R[-8]C:R[-1]C)"
    Range("A10:B10").Select
    Selection.Interior.ColorIndex = 36
    Range("B1").Select
    Selection.Interior.ColorIndex = 36
    Range("A2:A9").Select
    Selection.Interior.ColorIndex = 34
    Columns("A:A").ColumnWidth = 13.86
    Range("A2:B9").Select
    Charts.Add
    ActiveChart.ChartType = xlColumnClustered
    ActiveChart.SetSourceData _
    Source:=Sheets("Лист1").Range("A2:B9"), PlotBy:=xlColumns
    ActiveChart.Location Where:=xlLocationAsObject, Name:="Лист1"
    With ActiveChart
```

```
        .HasTitle = False
        .Axes(xlCategory, xlPrimary).HasTitle = False
        .Axes(xlValue, xlPrimary).HasTitle = False
    End With
End Sub
```

На первый взгляд полученный макрос выглядит довольно устрашающе, но прочитав данную книгу вы поймете, что его очень легко создавать и читать. Еще более замечательным является то, что в настоящий момент эта программа сама по себе совершенно не нужна и с ней можно работать даже не понимая записанные в ней коды. Пока о программе надо знать только ее имя — `Расходы` и то, что рабочий лист, на котором при помощи этого макроса будет строиться шаблон таблицы с диаграммой, должен иметь имя `Лист1`. Ограничение на выбор имени рабочего листа не столь обременительно. Изучив эту книгу, вы легко сможете создавать универсальные приложения без каких-либо ограничений на среду. Для того чтобы воспользоваться макросом, надо перед его выполнением переименовать рабочий лист, присвоив ему имя `Лист1`. После построения таблицы можно изменить имя рабочего листа на новое, например, на имя месяца, для которого строится текущий отчет по расходам.

Итак, активизируем новый рабочей лист, временно присвоим ему имя `Лист1`. Выберем команду **Сервис**, **Макрос**, **Макросы** (Tools, Macro, Macros), которая вызовет диалоговое окно **Макрос** (Macro). В этом окне в списке выделим исходный макрос и нажмем кнопку **Выполнить** (Run). Диалоговое окно закроется и выполнится процедура, создающая на активном рабочем листе шаблон таблицы. Теперь в нее остается ввести новые данные, а расчет суммарных расходов и построение диаграммы будет происходить автоматически.

Создание функций пользователя

VBA предоставляет также возможность пользователю создавать собственные функции, работать с которыми на рабочем листе можно при помощи мастера функций точно так же, как и с любой встроенной функцией. Рассмотрим один пример построения функции пользователя. На минуту представьте себе, что вы менеджер издательства по оптовой продаже книг. Для привлечения покупателей в вашем издательстве введена прогрессивная шкала цен. Если продается от 100 до 200 экземпляров книги, то скидка от ее отпускной цены составляет 7%, если продается от 201 до 300 экземпляров, то скидка составляет 10%, а если свыше 300 экземпляров — 15%. Кроме того, для постоянных клиентов предусмотрена дополнительная скидка в размере 5%. Создадим функцию пользователя с именем `Стоимость` для расчета стоимости партии книг. Аргументы этой функции назовем `ЦенаОднойКниги`, `Количество` и `Скидка`. Для аргумента `Скидка` предусмотрим только два допустимых зна-

чения: 1 для постоянных клиентов и 0 в противном случае. Построим пользовательскую функцию `Стоимость` следующим образом:

1. Выполните команду **Сервис**, **Макрос**, **Редактор Visual Basic** (Tools, Macro, Visual Basic Editor), чтобы открыть окно редактора Visual Basic.
2. Выполните команду **Вставка**, **Модуль** (Insert, Module) для создания листа модуля.
3. Выберите значок модуля в окне **Проект** (Project), чтобы активизировать окно редактора кода на листе модуля.
4. Наберите на листе модуля приведенную ниже процедуру.

```
Function Стоимость(ЦенаОднойКниги, Количество, Скидка)
'
' Вычисление стоимости без учета скидки для постоянных клиентов
'
If Количество < 100 Then
'
' Продажа до 99 экземпляров
'
    СтоимостьБезСкидки = ЦенаОднойКниги * Количество
    Else
    If Количество <= 200 Then
'
' Продажа от 100 до 200 экземпляров
'
        СтоимостьБезСкидки = ЦенаОднойКниги * Количество * 0.93
        Else
'
' Продажа от 201 до 300 экземпляров
'
        If Количество <= 300 Then
            СтоимостьБезСкидки = ЦенаОднойКниги * Количество * 0.9
            Else
'
' Продажа свыше 300 экземпляров
'
            СтоимостьБезСкидки = ЦенаОднойКниги * Количество * 0.85
        End If
    End If
End If
```

```
'
' Корректировка стоимости с учетом скидки для постоянных клиентов
'
If Скидка = 0 Then
    Стоимость = СтоимостьБезСкидки
Else
    Стоимость = СтоимостьБезСкидки * 0.95
End If
End Function
```

Итак, функция пользователя `Стоимость` создана. Она включена в категорию функций, определенных пользователем, мастера функций. Благодаря тому что в VBA допустимо применение русскоязычных имен, текст написанной программы ясен и прозрачен для понимания. Кроме того, это обеспечивает и простоту использования диалогового окна мастера функций для данной функции (рис. В.5). Названия всех параметров функции `Стоимость` в окне мастера функций выводятся также на русском языке. Доступная для понимания структура диалогового окна позволяет использовать функцию `Стоимость` любому пользователю, даже не владеющему VBA. Для ее применения достаточно знать только имя этой функции.

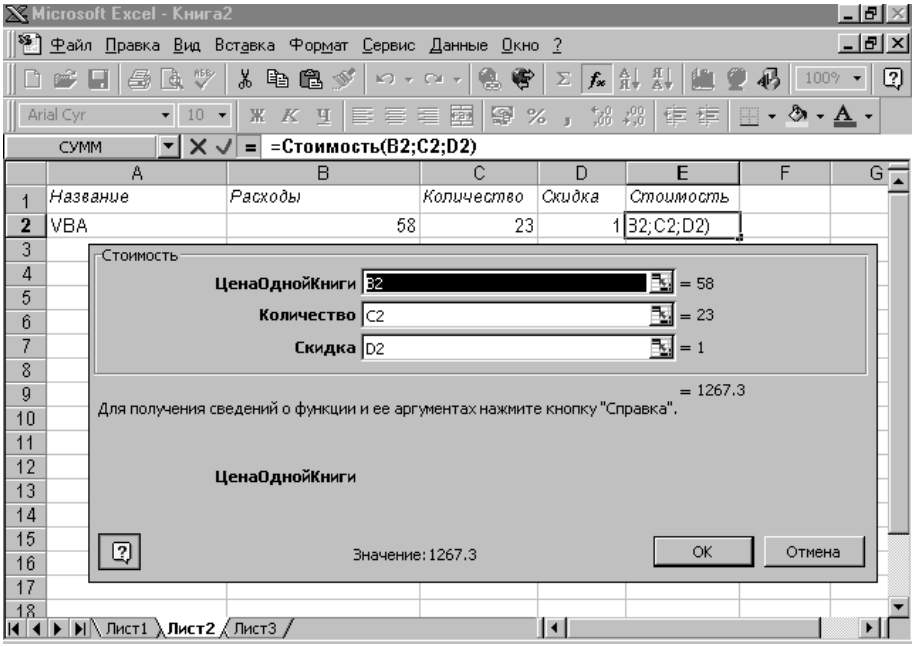
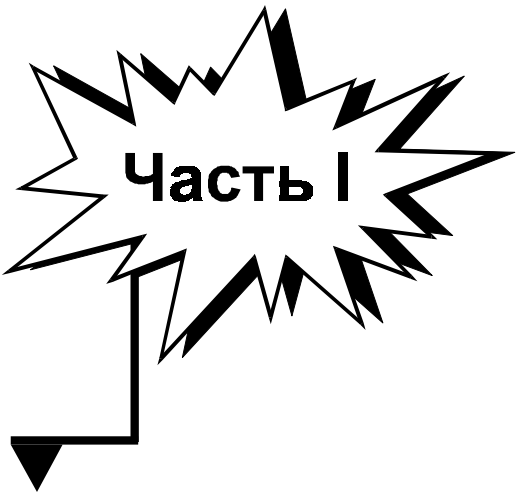
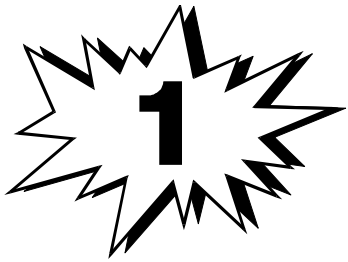


Рис. В.5. Диалоговое окно для заполнения параметров функции `Стоимость`



Основные средства и возможности VBA

Глава 1



Основные элементы VBA

Что такое VBA

VBA — относительно легкий язык программирования. Он прост в освоении и позволяет быстро получать ощутимые результаты — конструировать профессиональные приложения, решающие практически все задачи, встречающиеся в среде Windows. При этом создание многих приложений с использованием VBA проще и быстрее, чем при помощи других языков программирования.

VBA применяет технологию визуального программирования, т. е. конструирование рабочей поверхности приложения и элементов его управления непосредственно на экране, а также запись всей программы или ее частей при помощи MacroRecorder.

При беглом знакомстве с технологией визуального программирования может создаться впечатление, что она сводится к записи макросов, помещению в формы элементов управления и определении их связей с макросами. Довольно часто при решении простейших задач так и происходит. Но если требуется создать сколько-нибудь продвинутое приложение с содержательной обработкой данных, то на первый план выступает сам язык VBA. Такое заключение можно сделать даже на примере, рассмотренном в предыдущей главе. Если бы требовалось сделать макрос `Расходы` универсальным, не зависящим от имени рабочего листа, то необходимо было бы внести изменения в коды программы, а для этого надо понимать их структуру.

Объекты и их семейства

VBA относится к языкам объектно-ориентированного программирования (ООП). ООП можно описать как методику анализа, проектирования и написания приложений с помощью объектов. Что такое объект? Объект позволяет инкапсулировать данные вместе с кодом, предназначенным для их обработки, т. е. объединить их в нечто целое, именуемое объектом. VBA не

является объектно-ориентированном языком в строгом понимании этого слова, однако объектный подход играет в нем большую роль. Все визуальные объекты, такие как *рабочий лист (`Worksheet`), диапазон (`Range`), диаграмма (`Chart`), форма (`UserForm`),* являются объектами. В VBA имеется более 100 встроенных объектов.

Семейство (объект `Collection`) представляет собой объект, содержащий несколько других объектов, как правило, одного и того же типа. Например, объект `Workbooks` (рабочие книги) содержит все открытые объекты `Workbook` (рабочая книга). Каждый элемент семейства нумеруется и может быть идентифицирован либо по номеру, либо по имени. Например, `Worksheets(1)` обозначает первый рабочий лист активной книги, а `Worksheets("Лист1")` — рабочий лист с именем `Лист1`.

Объекты OLE и ActiveX

В VBA используется механизм OLE (Object Linking and Embedding — связывание и внедрение объектов), который позволяет взаимодействовать с любыми программами, поддерживающими OLE. Примером элементов, которые можно интегрировать при помощи механизма OLE, являются вставляемые объекты `OLEObject`, создаваемые, например, при помощи программ WordArt, ClipArt и т. д. Все OLE-объекты рабочего листа образуют семейство `OLEObjects`. Вручную в рабочий лист OLE-объекты вставляются командой **Вставка**, **Объект** (Insert, Object) с выбором в появившемся диалоговом окне **Вставка объекта** (Object) из списка на вкладке **Создание** (Create New) внедряемого объекта. OLE-объект отличается от обычного тем, что при выборе внедренного объекта (перемещении на него указателя и щелчке кнопкой мыши) активизируется программа, связанная с этим объектом, и меню приложения заменяется меню программы, его создавшей. Теперь можно, не выходя из основного приложения, работать с данным объектом, редактируя и видоизменяя его средствами создавшей его программы. Кроме того, OLE-технология обладает так называемым свойством `Automation`, с помощью которого можно устанавливать свойства, применять методы и обрабатывать события внедренных объектов, как обычных объектов приложения.

С 1996 года фирма Microsoft ввела новую терминологию и теперь то, что раньше именовалось OLE-объект, называется объектом ActiveX, а OLE Automation называется ActiveX Automation.

Классы

Важнейшим понятием ООП является *класс*. Класс обычно описывается, как проект, на основе которого впоследствии будет создан конкретный объект. Таким образом, класс определяет имя объекта, его свойства и действия, вы-

полняемые над объектом. В свою очередь каждый объект, в соответствии с описанным выше, является *экземпляром класса*.

Иерархия объектов

Объектная библиотека VBA располагает более 100 различных объектов, находящихся на различных уровнях иерархии. Иерархия определяет связь между объектами и показывает пути доступа к ним. На рис. 1.1 приведена модель встроенных объектов VBA.

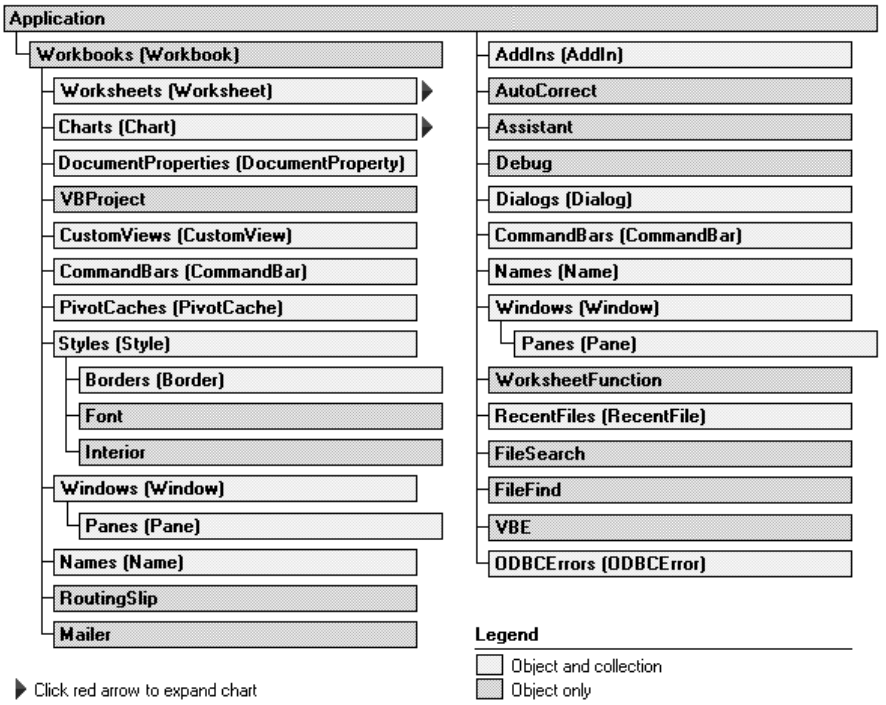


Рис. 1.1. Иерархия встроенных объектов VBA

Полная ссылка на объект состоит из ряда имен вложенных последовательно друг в друга объектов. Разделителями имен объектов в этом ряду являются точки, ряд начинается с объекта `Application` и заканчивается именем самого объекта. Например, полная ссылка на ячейку `A1` рабочего листа `Лист1` рабочей книги с именем `Архив` имеет вид:

```
Application.Workbooks("Архив").Worksheets("Лист1").Range("A1")
```

Приводить каждый раз полную ссылку на объект совершенно не обязательно. Обычно достаточно ограничиться только неявной ссылкой на объект.