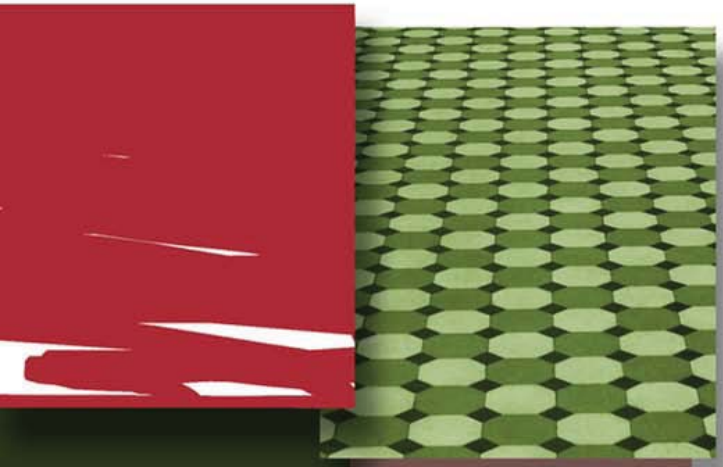
ВСЕВОЛОД НЕСВИЖСКИЙ

bhv®

1С: ПРЕДПРИЯТИЕ 8.0

ПРИЕМЫ ПРОГРАММИРОВАНИЯ



- ПОДРОБНОЕ ОПИСАНИЕ РАБОТЫ С БАЗОВЫМИ ОБЪЕКТАМИ
- АНАЛИЗ И ОБРАБОТКА ДАННЫХ
- ПОСТРОЕНИЕ ОТЧЕТОВ И ПЕЧАТНЫХ ФОРМ
- ВЗАИМОДЕЙСТВИЕ С ВНЕШНИМИ БАЗАМИ ДАННЫХ SQL
- ПРАВИЛА ПОСТРОЕНИЯ ЗАПРОСОВ

PRO

ПРОФЕССИОНАЛЬНОЕ ПРОГРАММИРОВАНИЕ

+CD

УДК 681.3.06
ББК 32.973.26-018.2
Н55

Несвижский В.

Н55 1С:Предприятие 8.0. Приемы программирования. — СПб.: БХВ-Петербург, 2007. — 512 с.: ил. + CD-ROM — (Профессиональное программирование)

ISBN 978-5-9775-0089-0

Книга полностью построена на реальных примерах и задачах, решаемых 1С-программистами в повседневной работе. Представленные приемы программирования универсальны и применимы в любых существующих конфигурациях системы 1С:Предприятие 8.0. Рассмотрены наиболее важные и часто используемые объекты конфигурации: документы, справочники, регистры накопления, регистры сведений, отчеты, макеты и др. Особое внимание уделено разработке печатных документов, применению построителя отчетов и анализу данных. Подробно описано, как подключить существующие внешние базы данных SQL к системе 1С:Предприятие 8.0. Отличительной особенностью книги является большое количество примеров с подробными комментариями. Исходные тексты всех примеров содержатся на прилагаемом компакт-диске.

Для 1С-программистов

УДК 681.3.06
ББК 32.973.26-018.2

Группа подготовки издания:

Главный редактор	*Екатерина Кондукова*
Зам. главного редактора	*Евгений Рыбаков*
Зав. редакцией	*Григорий Добин*
Редактор	*Елена Кашлакова*
Компьютерная верстка	*Натальи Караваевой*
Корректор	*Виктория Пиотровская*
Дизайн серии	*Инны Тачиной*
Оформление обложки	*Елены Беляевой*
Зав. производством	*Николай Тверских*

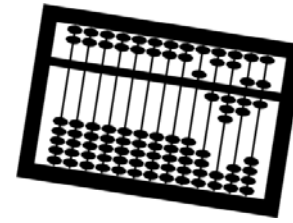
Лицензия ИД № 02429 от 24.07.00. Подписано в печать 22.06.07.
Формат $70\times100^{1}/_{16}$. Печать офсетная. Усл. печ. л. 41,28.
Тираж 2500 экз. Заказ №
"БХВ-Петербург", 194354, Санкт-Петербург, ул. Есенина, 5Б.

Санитарно-эпидемиологическое заключение на продукцию
№ 77.99.02.953.Д.006421.11.04 от 11.11.2004 г. выдано Федеральной службой
по надзору в сфере защиты прав потребителей и благополучия человека.

Отпечатано с готовых диапозитивов
в ГУП "Типография "Наука"
199034, Санкт-Петербург, 9 линия, 12

ISBN 978-5-9775-0089-0

Оглавление



Глава 4

Получение форм

Как известно, любое современное приложение, разрабатываемое для конечных пользователей, содержит в себе различные средства интерактивного взаимодействия с программой, иначе говоря, интерфейс. Система "1С:Предприятие 8.0" предоставляет разработчикам широкий выбор объектов, позволяющих быстро создавать полный набор интерфейсных модулей: меню, панелей инструментов, всевозможных элементов управления. К ним также относятся формы, играющие ключевую роль в формировании удобного интерфейса пользователя. Конфигуратор поддерживает создание общих форм (доступных из любого объекта конфигурации), форм справочников, форм документов, форм для отчетов и обработок, форм для журналов документов и регистров, а также для некоторых других объектов, не рассматриваемых в данной книге.

Документы, как правило, всегда содержат несколько форм. На этапе разработки конфигурации предоставляется возможность задавать для каждого документа различные типы форм. Наиболее часто используются несколько определенных типов форм:

- основная форма — позволяет пользователю создавать новый и корректировать существующий документ, заполнять реквизиты документа и табличных частей, сохранять документ в базе, проводить и отменять проводку;
- форма работы со списком — специальная форма, позволяющая отображать список однотипных документов, в которой доступны различные средства поиска и отбора, сортировки, создания и удаления, ввода на основании (создание нового документа другого типа на основании выбранного из списка) и многое другое;
- форма выбора — предоставляет возможность выбора из списка какого-либо документа с передачей по ссылке в другую форму.

Кроме перечисленных типов форм, имеется возможность создания пользовательских форм произвольного вида. В этом случае разработчик сам определяет предназначение и способы использования данных форм. Как правило, после формирования всех необходимых форм документа, одну из них назначают основной. Такая форма будет вызываться по умолчанию, если при вызове методов отображения не будет явно задан тип формы.

Здесь мы разберем приемы программирования только базовых и общих форм, но этого будет вполне достаточно для самостоятельной работы с любыми произвольными типами форм, поскольку для доступа и к тем и другим используются одни и те же базовые объекты: ДокументМенеджер, Документ Объект и ДокументСсылка.

В большинстве случаев, при разработке нового типа документа и создании для него различных форм, в качестве основной выбирают форму документа, которая содержит реквизиты и табличные части. Элементы управления, входящие в конфигуратор, позволяют в короткие сроки формировать удобный, красивый и функциональный интерфейс. Пример такого документа представлен на рис. 4.1.

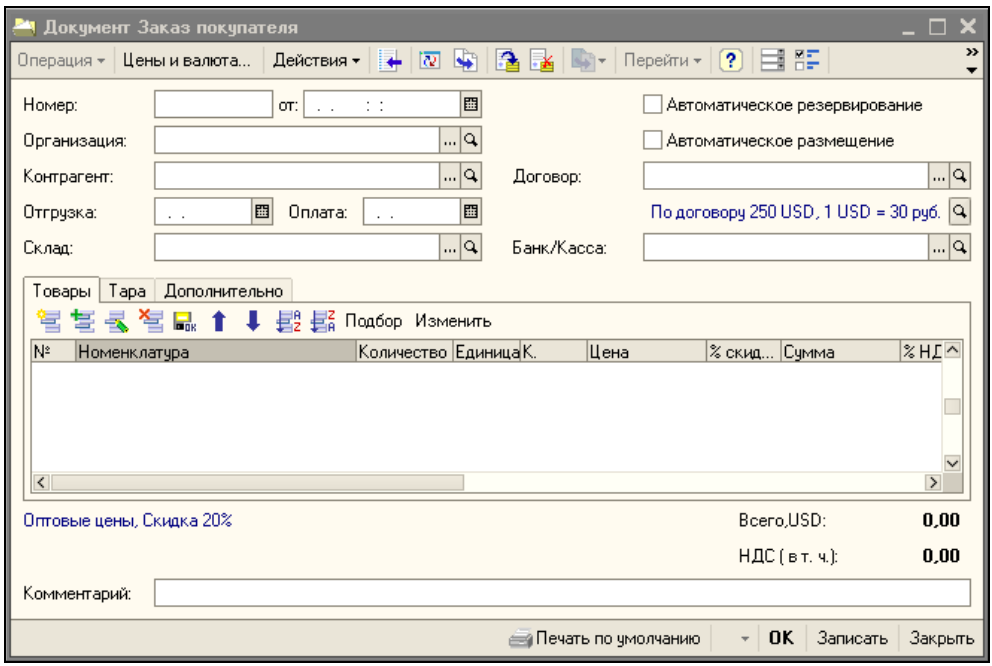


Рис. 4.1. Документ Заказ покупателя

Следует отметить, что большинство документов в системе "1С:Предприятие 8.0" имеют единый вид, что является примером грамотно продуманного интерфейса пользователя. Это дает возможность решить две наиболее важные задачи: удобство обучения и концентрация на самом процессе работы.

Для программного доступа к формам документа, как уже отмечалось ранее, служат несколько базовых объектов. Разберем их по порядку. Первый из них, ДокументМенеджер, предоставляет несколько удобных методов для получения объектов форм. С помощью метода `ПолучитьФорму` можно получить любую форму документа, заданную в конфигураторе на этапе проектирования. Посмотрите пример работы с данным методом, представленный в листинге 4.1.

Листинг 4.1. Получение форм документа

```
// Получаем новую форму документа
ФормаДокумента =
            Документы.ЗаказПокупателя.ПолучитьФорму("ФормаДокумента");
// Открываем полученную форму
ФормаДокумента.Открыть();

// Получаем форму выбора
ФормаВыбораДокумента =
            Документы.ЗаказПокупателя.ПолучитьФорму("ФормаВыбора");
// Проверяем, если форма уже открыта, просто активизируем ее
Если ФормаВыбораДокумента.Открыта() Тогда
      ФормаВыбораДокумента.Активизировать();
Иначе
      ФормаВыбораДокумента.ОткрытьМодально();
КонецЕсли;

// Получаем форму списка
ФормаСпискаДокумента =
            Документы.ЗаказПокупателя.ПолучитьФорму("ФормаСписка");
// Проверяем, если форма уже открыта, обновляем ее
Если ФормаСпискаДокумента.Открыта() Тогда
      ФормаСпискаДокумента.Обновить();
Иначе
      ФормаСпискаДокумента.Открыть();
КонецЕсли;
```

Как видно из примера, достаточно указать наименование формы документа, чтобы получить доступ ко всем свойствам и методам, управляющим поведением и отображением объекта формы. Для открытия формы мы использовали методы `Открыть` и `ОткрытьМодально`. Первый открывает форму и выводит ее на экран, а второй, кроме того, позволяет задать режим модальности. При таком режиме пользователь должен прежде закрыть данную форму, чтобы продолжить работу с другими окнами. Метод `Открыть` помогает убедиться, что открываемая форма уже не открыта раньше. Метода `Активизировать` делает открытую форму активной (доступной для ввода данных), а метод `Обновить` — просто обновляет все элементы формы.

Дополнительно, в методе `ПолучитьФорму` можно задать владельца открываемой формы, так называемую родительскую форму, из которой открывается данная форма. Владелец задается вторым необязательным аргументом. Существует еще и третий необязательный аргумент, представляющий собой уникальный ключ, по которому можно искать уже открытую форму. Для наглядности изложенного посмотрите пример, представленный в листинге 4.2.

Листинг 4.2. Дополнительные возможности метода `ПолучитьФорму`

```
// Получаем форму списка из другой формы-владельца
ФормаСпискаДокумента =
Документы.ОприходованиеТоваров.ПолучитьФорму("ФормаСписка", Владелец);
// Устанавливаем свойство закрытия при закрытии владельца формы
ФормаСпискаДокумента.ЗакрыватьПриЗакрытииВладельца = Истина;
// Открываем полученную форму
ФормаСпискаДокумента.Открыть();

// Получаем форму и задаем уникальный ключ
ФормаВыбораДокумента =
Документы.ОприходованиеТоваров.ПолучитьФорму("ФормаВыбора", , "ключ1");
// Открываем полученную форму
ФормаВыбораДокумента.Открыть();
// Получаем форму, используя уникальный ключ
ФормаВыбораДокумента =
Документы.ОприходованиеТоваров.ПолучитьФорму(, , "ключ1");
// Открываем форму в виде модального окна
ФормаВыбораДокумента.ОткрытьМодально();
```

В качестве владельца может выступать любая форма, заданная в конфигурации. Установленный признак `ЗакрыватьПриЗакрытииВладельца` приводит к автоматическому закрытию формы при закрытии владельца. Уникальный ключ, доступный как для чтения, так и для записи, помогает однозначно идентифицировать объект формы, исключив случайные совпадения имен. Неповторимое значение ключа можно сформировать с помощью объекта `УникальныйИдентификатор`. Использование ключа при открытии формы позволяет реализовать вывод на экран нескольких копий одного документа.

Для упрощения процесса вызова нужного типа формы, можно воспользоваться дополнительными методами объекта ДокументМенеджер:

- `ПолучитьФормуВыбора` — вызывает форму выбора для указанного типа документа;
- `ПолучитьФормуСписка` — вызывает форму списка документа;
- `ПолучитьФормуНовогоДокумента` — вызывает форму для заполнения нового документа.

Названия этих методов говорят сами за себя. Все они имеют по три необязательных аргумента: наименования формы, владельца формы и уникального ключа. В отличие от метода `ПолучитьФорму`, где первый аргумент (наименование формы, заданное в конфигураторе) требуется указывать всегда, дополнительные методы просто получают определенный тип формы. Пример использования этих методов представлен в листинге 4.3.

Листинг 4.3. Дополнительные методы получения форм документа

```
// Получаем форму списка документа
ФормаСпискаДокумента = Документы.АвансовыйОтчет.ПолучитьФормуСписка();
// Открываем полученную форму
ФормаСпискаДокумента.Открыть();

// Формируем уникальный ключ
ГУИД = Новый УникальныйИдентификатор;
// Получаем форму выбора документа и назначаем ей уникальный ключ
ФормаВыбораДокумента =
              Документы.АвансовыйОтчет.ПолучитьФормуВыбора(, , ГУИД);
// Открываем полученную форму
ФормаВыбораДокумента.ОткрытьМодально(10);
// Получаем эту же форму по значению ключа
ФормаВыбораДокументаПоКлючу =
```

```
                Документы.АвансовыйОтчет.ПолучитьФорму(, , ГУИД);
// Если форма уже открыта, активизируем ее
Если ФормаВыбораДокументаПоКлючу.Открыта() Тогда
        ФормаВыбораДокументаПоКлючу.Активизировать();
Иначе
        ФормаВыбораДокументаПоКлючу.Открыть();
КонецЕсли;

// Получаем форму нового документа
ФормаНовогоДокумента =
                Документы.АвансовыйОтчет.ПолучитьФормуНовогоДокумента();
```

Рассмотренные варианты получения различных типов форм демонстрируют широкие возможности, доступные разработчику для управления формами документов.

В базовом объекте ДокументОбъект имеется один метод для получения формы. Называется он `ПолучитьФорму`, имеет три необязательных аргумента, а в остальном ничем ни отличается от аналогичного метода объекта ДокументМенеджер. Для объекта ДокументСсылка также предусмотрен метод `ПолучитьФорму`. Примеры их использования приведены в листинге 4.4.

Листинг 4.4. Получение форм документа из объектов ДокументОбъект и ДокументМенеджер

```
// Делаем выборку документов из базы
ВыборкаПриходныйОрдерНаТовары =
                        Документы.ПриходныйОрдерНаТовары.Выбрать();
// Организуем цикл для обработки найденных документов
Пока ВыборкаПриходныйОрдерНаТовары.Следующий() Цикл
        // Получаем объект документа
        ОбъектПриходныйОрдерНаТовары =
                        ВыборкаПриходныйОрдерНаТовары.ПолучитьОбъект();
        // Получаем форму документа
        ФормаДокумента = ОбъектПриходныйОрдерНаТовары.ПолучитьФорму();
        Если НЕ ФормаВыбораДокументаПоКлючу.Открыта() Тогда
                ФормаДокумента.Открыть();
        КонецЕсли;
КонецЦикла;
```

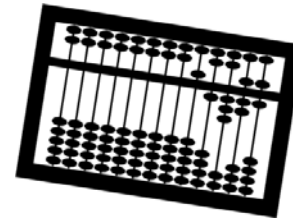
```
// Делаем выборку документов из базы
ВыборкаРеализацияТоваров = Документы.РеализацияТоваров.Выбрать();
// Организуем обход в цикле для всех полученных документов
Пока ВыборкаРеализацияТоваров.Следующий() Цикл
        // Получаем ссылку на документ
        СсылкаРеализацияТоваров = ВыборкаРеализацияТоваров.Ссылка;
        // Получаем форму документа
        ФормаДокумента = СсылкаРеализацияТоваров.ПолучитьФорму();
        // Открываем полученную форму документа
        ФормаДокумента.Открыть();
КонецЦикла;
```

И последнее, что хотелось бы представить вниманию читателей — это получение так называемых общих форм. Общие формы позволяют использовать какой-либо стандартный набор действий, доступный из любого объекта конфигурации. Например, чтобы не создавать для каждого документа формы поиска и отбора по списку, достаточно построить одну общую форму и вызывать по мере необходимости из разных документов. Для получения любой общей формы предназначен глобальный метод `ПолучитьОбщуюФорму`. Он имеет три необязательных аргумента, аналогичных рассмотренным ранее для метода `ПолучитьФорму`. Пример его использования представлен в листинге 4.5.

Листинг 4.5. Получение общих форм конфигурации

```
// Получаем стандартную общую форму ФормаЦеныИВалюта
ОбщаяФорма = ПолучитьОбщуюФорму("ФормаЦеныИВалюта");
// Выполняем настройку элементов управления общей формы
// . . .
// Открываем полученную форму
ОбщаяФорма.ОткрытьМодально();
```

Как видите, получить общую форму достаточно просто. Однако перед тем, как открыть ее, как правило, требуется выполнить настройку элементов управления и заполнить начальные параметры использования. Поскольку настройки даже для стандартных (поставляемых фирмой 1С) общих форм могут сильно отличаться, здесь мы их приводить не будем.



Глава 5

Чтение и установка свойств

В этой главе мы поговорим о существующих свойствах документов, доступных через базовые объекты конфигурации: ДокументОбъект, ДокументСписок, ДокументВыборка и ДокументСсылка. Свойства позволяют получить различную информацию о документе: значения реквизитов и табличных частей, определить состояние документа и многое другое. Установить свойства возможно только через ДокументОбъект, в остальных случаях доступно лишь чтение значений.

С помощью объекта ДокументОбъект, в первую очередь, можно обратиться к значениям реквизитов, заданным в конфигураторе для данного документа. Поскольку реквизиты у разных документов разные, необходимо изучить структуру каждого перед использованием. Реквизиты хранят различную информацию о документе: валюта, сумма документа, наименование организации и контрагента, вид документа, кратность, ответственный и др. Рассмотрим на примере, представленном в листинге 5.1, работу со свойствами документов.

Листинг 5.1. Чтение реквизитов документа

```
// Задаем критерии выборки по дате
НачальнаяДата = НачалоДня(ТекущаяДата());
КонечнаяДата  = КонецДня(ТекущаяДата());
// Делаем выборку документов из базы за текущий день
ВыборкаВнутреннийЗаказ =
Документы.ВнутреннийЗаказ.Выбрать(НачальнаяДата, КонечнаяДата);
// Организуем цикл для обработки найденных документов
Пока ВыборкаВнутреннийЗаказ.Следующий() Цикл
```

```
        // Получаем объект документа
        Объект = ВыборкаВнутреннийЗаказ.ПолучитьОбъект();
        // Читаем значения реквизитов и выводим в окно сообщений
        Сообщить("Номер документа: " + Объект.Номер);
        // Валюта документа
        Сообщить("Валюта документа: " + Объект.ВалютаДокумента);
        // Наименование склада компании
        Сообщить("Склад компании: " + Объект.СкладКомпании);
        // Признак автоматического резервированиия документа
        Сообщить("Авторезервирование: " +
                    ?(Объект.АвтоРезервирование, "истина", "ложь");
        // Признак автоматического размещения
        Сообщить("Авторазмещение: " +
                    ?(Объект.АвтоРазмещение, "истина", "ложь");
        // Наименование организации
        Сообщить("Наименование организации: " +
                                            Объект.Организация);
        // Ответственный за формирование документа
        Сообщить("Ответственный сотрудник: ", Объект.Ответственный);
КонецЦикла;
```

Как видно из примера, вначале мы организовали выборку документов на текущую дату. Для получения объекта применили метод `ПолучитьОбъект`. Таким образом, нам стали доступны все реквизиты документа, которые мы успешно прочитали и вывели в окно служебных сообщений 1С.

Установка реквизитов не сильно отличается от чтения, но следует правильно передавать значения типов, согласно назначенным в конфигураторе. Пример записи реквизитов представлен в листинге 5.2.

Листинг 5.2. Установка реквизитов документа

```
// Задаем критерии выборки по дате
НачальнаяДата = НачалоДня(ТекущаяДата());
КонечнаяДата  = КонецДня(ТекущаяДата());
// Делаем выборку документов из базы за текущий день
ВыборкаЗаказПоставщику =
Документы.ЗаказПоставщику.Выбрать(НачальнаяДата, КонечнаяДата);
// Организуем цикл для обработки найденных документов
```

```
СчетчикНомера = 1;
Пока ВыборкаЗаказПоставщику.Следующий() Цикл
        // Получаем объект документа
        Объект = ВыборкаЗаказПоставщику.ПолучитьОбъект();
        // Устанавливаем значения реквизитов
        Объект.Номер = "Новый номер " + Строка(СчетчикНомера);
        // Получаем ссылку на валюту
        ССылкаВалюта = Справочники.Валюты.НайтиПоНаименованию("RUR");
        // Устанавливаем выбранную валюту для документа
        Объект.ВалютаДокумента = ССылкаВалюта;
        // Выбираем новую организацию
        ССылкаОрганизация = Справочники.Организации.НайтиПоКоду("0015");
        // Устанавливаем новую ссылку на организацию
        Объект.Организация = ССылкаОрганизация;
        // Выбираем нового ответственного
        ССылкаОтветственный =
        Справочники.ФизическиеЛица. НайтиПоНаименованию("Иванов И.И.");
        // Сохраняем сделанные изменения в базе
        Объект.Записать();
        СчетчикНомера = СчетчикНомера + 1;
КонецЦикла;
```

Итак, в рассмотренном примере мы выполнили установку реквизитов документов. Список документов получили через объект ДокументВыборка. Во-первых, мы изменили номера выбранных документов, воспользовавшись числовой переменной `СчетчикНомера` для формирования уникального значения. Далее установили новую валюту. Поскольку тип реквизита является ссылкой на справочник валют, нам понадобилось вначале получить ссылку на выбранную валюту. Для этого был выбран метод `НайтиПоНаименованию`. Как он работает, вы уже знаете. Кроме того, используя ссылки, мы скорректировали в документах организацию и ответственного за создание документа. Чтобы сохранить сделанные изменения, вызвали метод `Записать`.

Объект ДокументОбъект позволяет также получить доступ к табличным частям документа. Имена табличных частей, заданные в конфигураторе, и являются свойствами документа. Однако, в отличие от реквизитов, данные табличных частей можно только читать. Посмотрите, как это делается в листинге 5.3.

Листинг 5.3. Чтение реквизитов табличных частей документа

```
// Делаем выборку всех документов из базы
ВыборкаРеализацияТоваровУслуг =
                              Документы.РеализацияТоваровУслуг.Выбрать();
// Организуем цикл для обработки найденных документов
Пока ВыборкаРеализацияТоваровУслуг.Следующий() Цикл
        // Получаем объект документа
        Объект = ВыборкаРеализацияТоваровУслуг.ПолучитьОбъект();
        // Получаем таблиную часть Товары
        ТЧ_Товары = Объект.Товары;
        // Выводим наименование номенклатуры
        Сообщить("Номенклатура: " + ТЧ_Товары.Номенклатура);
        // Выводим характеристику номенклатуры
        Сообщить("Характеристика номенклатуры: " +
                              ТЧ_Товары.ХарактеристикаНоменклатуры);
        // Выводим количество товара
        Сообщить("Количество: " + Строка(ТЧ_Товары.Количество));
        // Выводим серию номенклатуры
        Сообщить("Серия номенклатуры: " + ТЧ_Товары.СерияНоменклатуры);
        // Получаем табличную часть Услуги
        ТЧ_Услуги = Объект.Услуги;
        // Получаем наименование услуги
        Сообщить("Наименование услуги: " + ТЧ_Услуги.Номенклатура);
        // Получаем количество
        Сообщить("Количество: " +
               Формат(ТЧ_Услуги.Количество, "ЧЦ=15;ЧДЦ=3"));
        // Получем единицу измерения
        Сообщить("Единица измерения: " +
               Строка(ТЧ_Услуги.ЕдиницаИзмерения));
        // Получаем табличную часть ВозвратнаяТара
        ТЧ_ВозвратнаяТара = Объект.ВозвратнаяТара;
        // Получаем наименование возвратной тары
        Сообщить("Наименование тары: " + ТЧ_ВозвратнаяТара.Номенклатура);
        // Получаем количество
        Сообщить("Количество: " +
               Формат(ТЧ_ВозвратнаяТара.Количество, "ЧЦ=15;ЧДЦ=3"));
        // Получаем цену
        Сообщить("Цена: " +
```

```
            Формат(ТЧ_ВозвратнаяТара.Цена, "ЧЦ=15;ЧДЦ=2"));
        // Получаем значение суммы
        Сообщить("Сумма возвратной тары: " +
            Формат(ТЧ_ВозвратнаяТара.Сумма, "ЧЦ=15;ЧДЦ=2"));
КонецЦикла;
```

Как видите, получить значения реквизитов табличных частей достаточно просто. Поскольку документ РеализацияТоваровУслуг в варианте конфигурации УПП содержит три табличные части, мы получили все три. Однако в вашей конфигурации может отсутствовать, например, табличная часть Услуги, поэтому всегда проверяйте структуру документов именно в той конфигурации, в которой вы работаете.

Объект ДокументОбъект позволяет еще управлять следующими значениями свойств: номер и дата документа, признак проведения и признак удаления. Эти свойства доступны как для чтения, так и для записи. Пример работы с ними представлен в листинге 5.4.

Листинг 5.4. Чтение и установка свойств документа

```
// Делаем выборку всех документов из базы
ВыборкаПоступлениеДопРасходов =
                        Документы.ПоступлениеДопРасходов.Выбрать();
// Организуем цикл для обработки найденных документов
Пока ВыборкаПоступлениеДопРасходов.Следующий() Цикл
        // Получаем объект документа
        Объект = ВыборкаПоступлениеДопРасходов.ПолучитьОбъект();
        // Получаем и устанавливаем номер документа
        НомерДокумента = Объект.Номер;
        Объект.Номер = НомерДокумента + " новый";
        // Получаем и устанавливаем дату документа
        ДатаДокумента = Объект.Дата;
        Объект.Дата = ТекущаяДата();
        // Определяем признак проведения документа
        Если Объект.Проведен Тогда
            Объект.Проведен = Ложь;
        КонецЕсли;
        // Определяем признак пометки на удаление
        Если Объект.ПометкаУдаления Тогда
            Объект = Ложь;
```

```
        КонецЕсли;
        // Сохраняем изменения в базе
        Объект.Сохранить();
КонецЦикла;
```

И последние свойства, о которых хотелось бы здесь упомянуть, доступны только для чтения и помогают получить сведения о движениях документа, ссылке на документ и обращения к самому объекту документа.

Движения документа можно прочитать из свойства `Движения`, которое возвращает коллекцию наборов записей о движениях документа в системе. На этапе разработки в свойствах документа задаются объекты (регистры), по которым документ может выполнять движения. Приведем пример получения движений документа по заданным в конфигураторе регистрам (листинг 5.5).

Листинг 5.5. Получение информации о движениях документа

```
// Делаем выборку всех документов из базы
ВыборкаПланПродаж = Документы.ПланПродаж.Выбрать();
// Организуем цикл для обработки найденных документов
Пока ВыборкаПланПродаж.Следующий() Цикл
        // Получаем объект документа
        Объект = ВыборкаПланПродаж.ПолучитьОбъект();
        // Получаем коллекцию движений документа
        ДвиженияПоРегистрам = Объект.Движения;
        // Выбираем движения по регистру сведений ЦеныКонтрагентов
        ДвиженияЦеныКонтрагентов = ДвиженияПоРегистрам.ЦеныКонтрагентов;
        // Читаем набор записей
        ДвиженияЦеныКонтрагентов.Прочитать();
        Для Каждого Запись Из ДвиженияЦеныКонтрагентов Цикл
                Сообщить(Запись.Цена);
        КонецЦикла;
        // Выбираем движения по регистру накопления ЗаказыПокупателей
        ДвиженияЗаказыПокупателей = ДвиженияПоРегистрам.ЗаказыПокупателей;
        // Читаем набор записей
        ДвиженияЗаказыПокупателей.Прочитать();
        Для Каждого Запись Из ДвиженияЗаказыПокупателей Цикл
                Сообщить(Запись.Количество);
        КонецЦикла;
```

```
        // Выбираем движения по регистру накопления ПартииТоваровКомпании
        ДвиженияПартииТоваровКомпании =
                        ДвиженияПоРегистрам.ПартииТоваровКомпании;
        // Читаем набор записей
        ДвиженияПартииТоваровКомпании.Прочитать();
        Для Каждого Запись Из ДвиженияПартииТоваровКомпании Цикл
                Сообщить(Запись.Количество);
                Сообщить(Запись.Стоимость);
        КонецЦикла;
КонецЦикла;
```

Итак, сделав выборку из базы, мы получили доступ к объекту документа. После этого, используя свойство `Движения`, прочитали коллекцию наборов движений документа по регистрам. По каким именно регистрам проходит документ, задается в конфигураторе. В данном случае, мы выбрали данные о движениях по регистру сведений ЦеныКонтрагентов и двум регистрам накопления — ЗаказыПокупателей и ПартииТоваровКомпании. С помощью метода `Прочитать` получаем из регистра набор записей для указанного документа. Далее делаем обход по записям и считываем значения ресурсов регистра.

Свойство `Ссылка` позволяет получить ссылку на документ, а свойство `ЭтотОбъект` возвращает сам объект документа. Его удобно применять в модулях формы или объекта. Пример работы с этими свойствами представлен в листинге 5.6.

Листинг 5.6. Использование свойств `Ссылка` и `ЭтотОбъект`

```
// Делаем выборку всех документов из базы
ВыборкаЗаказПоставщику = Документы.ЗаказПоставщику.Выбрать();
// Организуем цикл для обработки найденных документов
Пока ВыборкаЗаказПоставщику.Следующий() Цикл
        // Получаем объект документа
        Объект = ВыборкаЗаказПоставщику.ПолучитьОбъект();
        // Получаем ссылку на документ
        ССылкаЗаказПоставщику = Объект.Ссылка;
КонецЦикла;

// Если мы находимся в модуле объекта или формы документа
// Получаем ссылку на документ без метода ПолучитьОбъект
ССылкаЗаказПоставщику = ЭтотОбъект.Ссылка;
```

Объект ДокументСписок также имеет несколько полезных свойств, о которых мы сейчас поговорим. К слову, создать данный объект программно не получится. Он создается системой автоматически на этапе создания формы списка документов или явном добавлении разработчиком на закладке реквизитов формы. С помощью него организуется отбор, сортировка и размещение колонок в таблице списка документов. Через свойства этого объекта можно управлять размещением и отображением колонок в табличной части, работать с отборами и сортировкой. Отборы нужны для наложения ограничений (по дате, организации и т. д.) на отображаемую в таблице информацию. Сортировки помогают упорядочить вывод данных по порядку (например, по дате создания в порядке убывания). Посмотрите в листинге 5.7, как можно применять свойства объекта ДокументСписок для вывода информации об установленных полях отбора.

Листинг 5.7. Получение данных об установленных отборах

```
// Прочитаем отборы, установленные для списка документов формы
Для Каждого ЭлементОтбора Из ЭтаФорма.ДокументСписок.Отбор Цикл
        // Проверяем использование поля в отборе
        Если ЭлементОтбора.Использование Тогда
                // Выводим имя и представление элемента отбора
                Сообщить(ЭлементОтбора.Имя + ЭлементОтбора.Представление);
                // Выводим вид сравнения
                Сообщить(ЭлементОтбора.ВидСравнения);
                // Выводим значение отбора
                Сообщить(ЭлементОтбора.Значение);
                // Выводим диапазон значений
                Сообщить(ЭлементОтбора.ЗначениеС + " " +
                                                ЭлементОтбора.ЗначениеПо);
        КонецЕсли;
КонецЦикла;
```

Для получения данных полей отбора мы применили цикл, в котором выбрали с помощью объекта Отбор имена и представления всех используемых (свойство `Использование`) отборов, а также вид сравнения и выбранный период. Чтобы установить отбор по определенному полю таблицы, достаточно через объект Отбор присвоить значение поля отбора и установить для него свойство `Использование` в значение `Истина`. Пример установки отбора по реквизитам документа представлен в листинге 5.8.