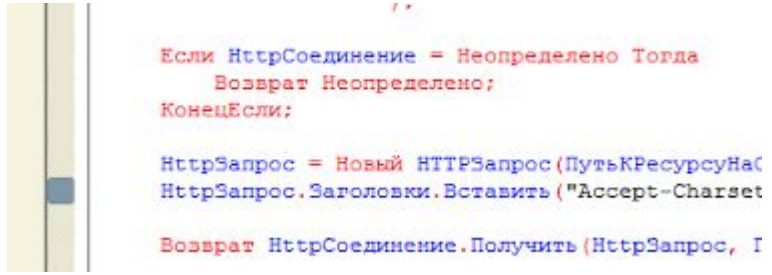
МДК 05.02 Разработка кода информационных систем

ФИРМА "1С"

Назначение и краткая характеристика языка программирования 1С. Виды программных модулей

Встроенный язык системы 1С: Предприятие предназначен для описания алгоритмов функционирования той или иной прикладной задачи. Встроенный язык представляет собой предметно-ориентированный язык программирования высокого уровня, разработанный специально с учетом возможности его применения не только опытными программистами. В частности, все операторы языка имеют как русское, так и англоязычное написание, которые можно комбинировать в одном исходном тексте.



```
Если HttpСоединение = Неопределено Тогда
    Возврат Неопределено;
КонецЕсли;

HttpЗапрос = Новый HTTPЗапрос(ПутьКРесурсуНа(
HttpЗапрос.Заголовки.Вставить("Accept-Charset

Возврат HttpСоединение.Получить(HttpЗапрос, I
```

```
Function PervyiBlinKomom(EtoPervyiBlin)

    If EtoPervyiBlin = True Then
        Return True;
    Else
        Return False;
    EndIf;

EndFunction
```

Ключевым понятием всей системы, является объект. Объект – это совокупность свойств (т.е. данных, характерных для этого объекта), методов их обработки (подпрограмм изменения свойств) и событий, на которые объект может реагировать и которые приводят, как правило, к изменению его свойств. **Любой создаваемый объект будет принадлежать или константам, или справочникам, или документам и т.д.**

Виды программных модулей в системе 1С: Предприятие

Программные модули в конфигурации не являются самостоятельными программами. Это связано с тем, что они только часть всей конфигурации задачи.

То есть программный модуль – это своего рода «контейнер» для размещения текстов процедур и функций, вызываемых системой только во время возникновения тех или иных событий, вызванных со стороны пользователя (значит» только в строго определенные моменты времени). Поэтому программный модуль не имеет формальных границ своего описания типа «начало модуля» – «конец модуля». Его границами фактически являются границы того текстового документа, в котором размещен модуль.

Место размещения программного модуля предоставляется Конфигуратором в тех местах конфигурации прикладной задачи, которые могут потребовать описания специфических алгоритмов функционирования.

Важное следствие отсюда – мы не можем создавать какие-либо произвольные модули. Все алгоритмы следует оформлять в виде процедур или функций, которые будут вызваны системой в заранее предусмотренных ситуациях (например, при нажатии кнопки, при открытии формы документа, при вводе нового элемента справочника и т.д.).

Виды программных модулей в системе 1С: Предприятие

Приведем таблицу, в которой перечислим все виды программных модулей, а кроме того, такие важнейшие их атрибуты, как расположение в дереве метаданных и момент выполнения (запуска)

Наименование модуля	Расположение	Запуск
Глобальный модуль	Метаданные → Глобальный модуль	Запускается при начале работы всей прикладной задачи
Модуль формы списка справочника	Метаданные → Справочник → Форма списка	Запускается при открытии формы списка справочника
Модуль формы группы справочника	Метаданные → Справочник → Форма группы	Запускается при открытии формы группы справочника
Модуль формы элемента справочника	Метаданные → Справочник → Форма	Запускается при открытии формы элемента справочника
Модуль формы документа	Метаданные → Документ →Форма	Запускается при открытии формы документа

Виды программных модулей в системе 1С: Предприятие

Приведем таблицу, в которой перечислим все виды программных модулей, а кроме того, такие важнейшие их атрибуты, как расположение в дереве метаданных и момент выполнения (запуска)

Наименование модуля	Расположение	Запуск
Модуль документа	Метаданные → Документ → Модуль документа	Запускается при проведении документа, при удалении проведенного документа из журнала, при снятии проведения, при выполнении архивации записей журнала расчетов, порожденных этим документом
Модуль формы журнала документов	Метаданные → Журнал документов → Форма	Запускается при открытии формы журнала документов
Модуль формы журнала расчетов	Метаданные → Журнал расчетов → Форма	Запускается при вызове формы журнала расчетов
Модуль формы списка счетов (плана счетов)	Метаданные → План счетов	Запускается при вызове формы списка счетов

Виды программных модулей в системе 1С: Предприятие

Наименование модуля	Расположение	Запуск
Модуль формы счета	Метаданные → Планы счетов → Форма счета	Запускается при открытии формы счета
Модуль формы журнала операций	Метаданные → Журнал операций → Форма	Запускается при вызове формы журнала операций
Модуль формы операции	Метаданные → Операция	Запускается при вызове формы журнала операций
Модуль формы журнала проводок	Метаданные → Журнал проводок → Форма	Запускается, при вызове формы журнала проводок
Модуль формы отчета	Метаданные → Отчет → Форма	Запускается при открытии диалоговой формы подготовки отчета
Модуль формы обработки	Метаданные → Обработка → Форма	Запускается при открытии диалоговой формы обработки
Модуль вида расчета	Метаданные → Вид расчета → Модуль вида расчета	Запускается при расчете соответствующих записей журнала расчетов

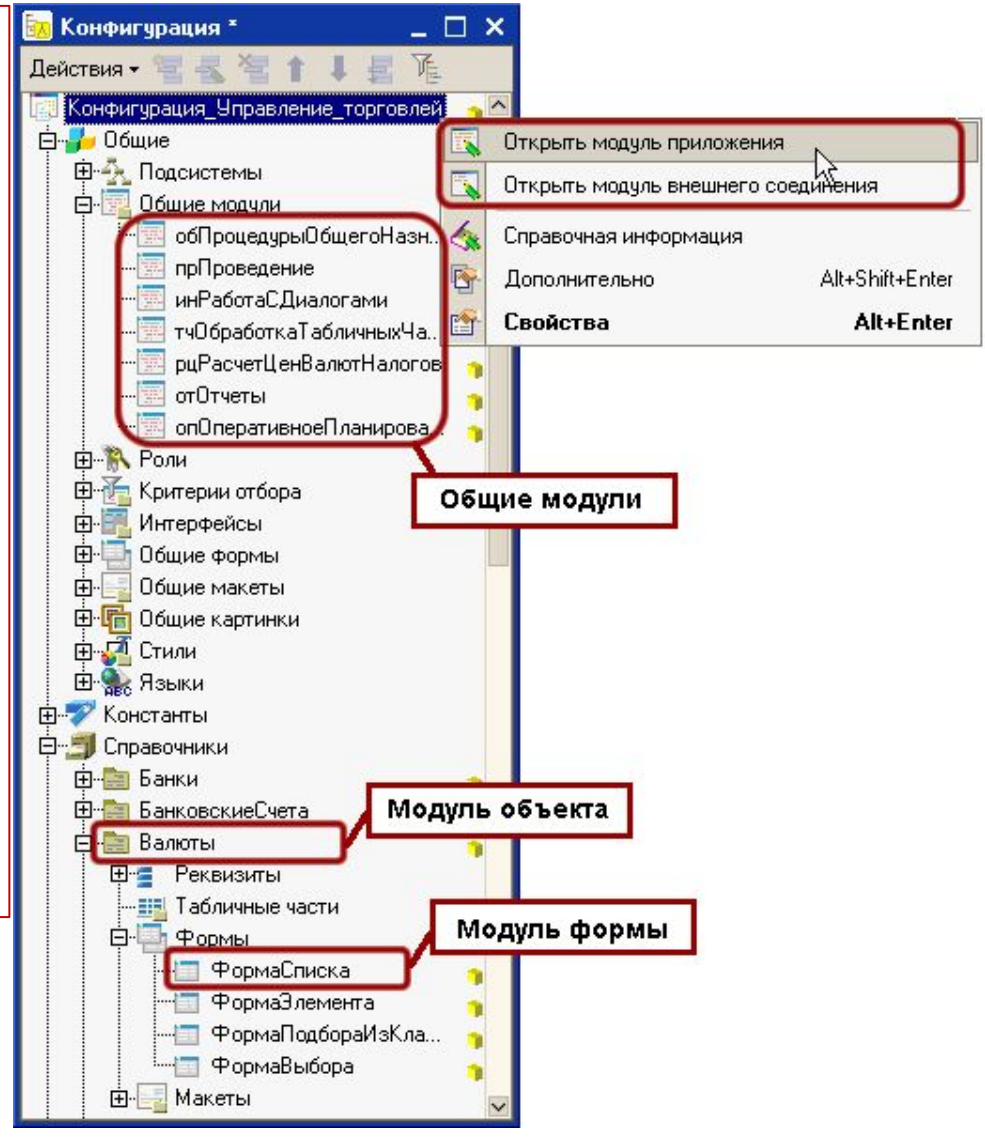
Из таблицы следует, что фактически существуют два типа модулей: модуль формы объекта и модуль объекта.

Виды программных модулей в системе 1С: Предприятие

Глобальный модуль позволяет расширить и дополнить функциональные возможности 1С: Предприятие в соответствии со спецификой конкретной прикладной задачи.

Если такая настройка не требуется, то глобальный Модуль формы объекта изначально служит для достижения двух основных целей: описания алгоритма взаимодействия объекта с пользователем и формирования того или иного печатного представления объекта. Некоторые из этих алгоритмов для какого-то объекта могут и отсутствовать.

Модули формы существуют практически для всех типов объектов.

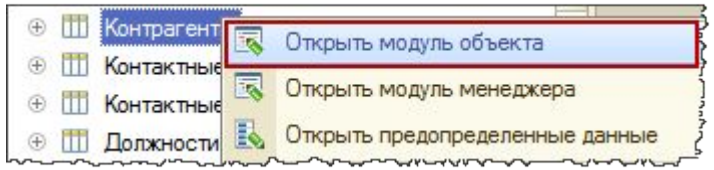


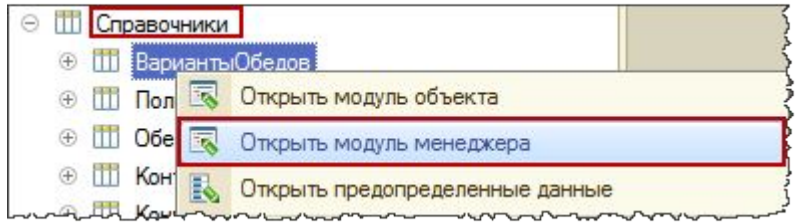
Модуль объекта присутствует только у двух типов объектов метаданных, а именно **документа и вида расчета.**

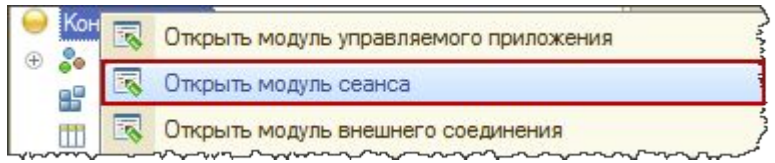
Виды программных модулей в системе 1С: Предприятие

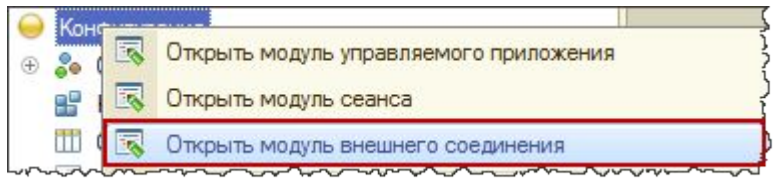
Модуль документа содержит предопределенные процедуры. Они описывают алгоритм обработки реквизитов и осуществления движений по счетам, регистрам учета и т.д. в процессе проведения документа (а также при удалении проведения, снятии проведения, архивации записей журнала расчетов).

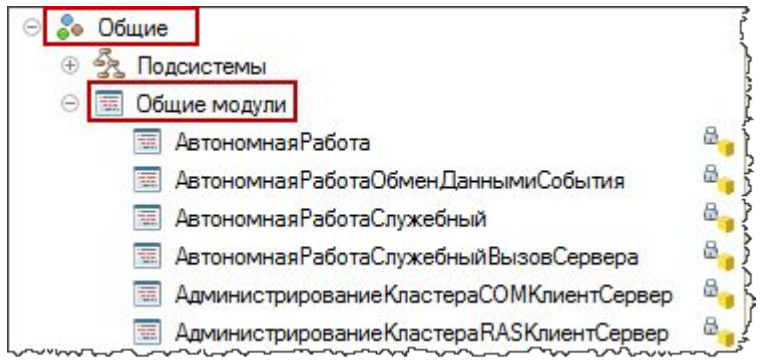
Модуль расчета содержит алгоритм выполнения данного вида расчета, он же заменяет модуль формы, т.к. таковой просто отсутствует у вида расчета.











Формат программного модуля

Неотъемлемой частью любого программного модуля являются **комментарии.** Комментарии служат для всякого рода пояснений работы модуля. К тому же систематическое комментирование программы является признаком хорошего тона в программировании. В тексте модуля комментарий начинается с пары символов // и заканчивается концом строки. Следовательно, удобно располагать комментарий в строке кода, после него, или же возможна целая строка, состоящая только из комментария. После символов // **операторы располагать нельзя**, т.к. они будут расценены компилятором как часть комментария.

Пример:

```
Контр = док.Контрагент;    //в переменную контр будет
                           //скопировано наименование контрагента
//это тоже комментарий, занимающий всю строку до конца
```

Формат программного модуля

Затронем два важных понятия языка: **константы и переменные**.

Константа – это постоянная величина, значение которой не может изменяться во время выполнения программы. Тип константы соответствует некоторому базовому типу. Следовательно, константы подразделяются на:

- числовые – это не что иное, как десятичное число, *например: 235, -5, 54.8;*
- строковые – это произвольные последовательности символов, заключенные в кавычки, например: "Это и есть строковая константа";
- константы типа «дата» – это взятые в одинарные кавычки три двухразрядных числа, разделенных точками, например: '2003.04.23', '1999.10.01'. Для записи года разрешается использовать четыре цифры: '1995.02.15';
- системные строковые константы. К ним относятся РазделительСтраниц, РазделительСтрок и СимволТабуляции.

Формат программного модуля

Затронем два важных понятия языка: **константы и переменные**.

Переменная – величина, значение которой допускается изменять в процессе выполнения программы. Она имеет свой идентификатор, удовлетворяющий правилам формирования переменных. Явно объявлять переменные не обязательно. Объявлением переменной считается ее первое использование в левой части оператора присваивания. Любая переменная обязательно должна быть проинициализирована перед ее использованием в правой части оператора присваивания.

Обращение в программах к объектам, строкам и т.д., осуществляется посредством **имен (идентификаторов) переменных**. Именем переменной, процедуры или функции может быть любая последовательность букв, знаков подчеркивания «_» и цифр, начинающаяся с буквы или со знака подчеркивания. Как и в любом языке программирования, вновь создаваемые имена не должны совпадать с теми, которые уже существуют; кроме того, они должны отличаться и от зарезервированных слов языка, имен существующих процедур и функций. *Регистр букв значения не имеет. Переменная абв – это то же самое, что и АБВ, АбВ, абВ и т.д.*

Формат программного модуля

В любом языке программирования существуют зарезервированные слова. Это такие слова, которые не могут использоваться в качестве имен переменных и процедур (функций) и несут свою предопределенную смысловую нагрузку. В языке 1С они имеют два представления – русское и английское. Как и в именах переменных, регистр букв не учитывается.

Полный список ключевых слов встроенного языка в обоих вариантах представления (русско- и англоязычном).

Формат программного модуля

Если	If	Или	Or
Тогда	Then	Не	Not
ИначеЕсли	ElsIf	Знач	Val
Иначе	Else	СтрДлина	StrLen
КонецЕсли	EndIf	СокрЛ	TrimL
Цикл	Do	СокрП	TrimR
Для	For	Лев	Left
По	To	Прав	Right
Пока	While	Сред	Mid
КонецЦикла	EndDo	Цел	Int
Процедура	Procedure	Окр	Round
Функция	Function	Число	Number
КонецПроцедуры	EndProcedure	Строка	String
КонецФункции	EndFunction	Дата	Date
Перем	Var	Формат	Format
Перейти	Goto	Разм	Dim
Возврат	Return	Вопрос	DoQueryBox
Продолжить	Continue	Предупреждение	DoMessageBox
Прервать	Break	Контекст	Context
И	And		

Формат программного модуля

Любой программный модуль 1С: Предприятия имеет одну и ту же структуру. Программный модуль состоит из следующих разделов:

- **раздел определения переменных;**
- **раздел описания процедур и функций;**
- **раздел выполняемой части, т.е. основной программы.**

Надо сказать, что любой из этих разделов в каком-либо отдельном модуле может отсутствовать. Важен только порядок их расположения, сначала всегда следует определение переменных, затем описание процедур и функций и, наконец, основная программа. Какой-либо явной границы между разделами не существует, но она очень легко определяется самостоятельно по смыслу. Дело в том, что текст программы представляет собой последовательность операторов, разделенных символом «;». Поэтому признаком начала нового раздела (и окончания предыдущего) является тип следующего оператора.

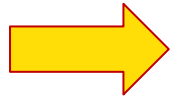
Формат программного модуля

Раздел описания переменных содержит в себе операторы объявления переменных Перем, разделенные символом «;». Заканчивается он с началом описания первой процедуры или функции (словом Процедура или Функция с последующим идентификатором таковой).

Раздел описания процедур и функций заканчивается с первым исполняемым оператором основной программы. Как правило, он самый большой по количеству строк кода, ведь именно в нем содержатся описания созданных нами алгоритмов.

Раздел основной программы завершает программный модуль. Интересно, что он отправляется на выполнение сразу же в момент запуска модуля. Этот раздел может содержать только исполняемые операторы. Наиболее правильно применять его для инициализации некоторыми требуемыми нам значениями переменных, которые могут быть необходимы перед использованием в процедурах или функциях модуля.

Специальные символы встроенного языка



Написание символа	Значение символа
//	Начинает комментарий. Комментарий продолжается до конца строки
;	Символ разделения операторов
( )	В круглые скобки мы заключаем список параметров процедур, функций и методов
,	Запятой разделяют параметры в списке параметров функций, процедур, методов
|	Символ переноса строки. Используется только в начале строки и означает, что данная строка есть продолжение предыдущей
[ ]	В квадратных скобках задается размерность массива
« »	В двойные кавычки заключаются строковые константы
‘ ’	В одинарные кавычки заключаются константы типа «Дата»
.	Десятичная точка в константах типа «Число». Служит также разделителем при обращении к атрибутам и методам объектов
?	Системная процедура «условное выполнение»
:	Двоеточие служит последним символом в имени метки
=	Логическая операция «равно» или символ присвоения значения
+	Знак операции арифметического сложения. Знак конкатенации строк
-	Знак операции арифметического вычитания. Унарный минус
*	Знак операции умножения
/	Операция деления
%	Знак «процент» означает остаток от деления
>	Знак логической операции «больше»
<	Знак логической операции «меньше»
>=	Знак логической операции «больше или равно»
<=	Знак логической операции «меньше или равно»
≠	Знак логической операции «не равно»
~	Тильда. Этим знаком начинается метка оператора

Типы данных системы

В любом языке программирования существует свой набор типов данных. Их обычно классифицируют по таким признакам, как *набор допустимых операций*, *вид сохраняемой информации* (числовые типы, строковые и т.д.), *диапазон возможных значений*.

Во встроенном языке программирования системы 1С: Предприятие типы данных принято подразделять на базовые (элементарные) и агрегатные.

С точки зрения расширяемости, агрегатные типы данных делятся на две большие категории:

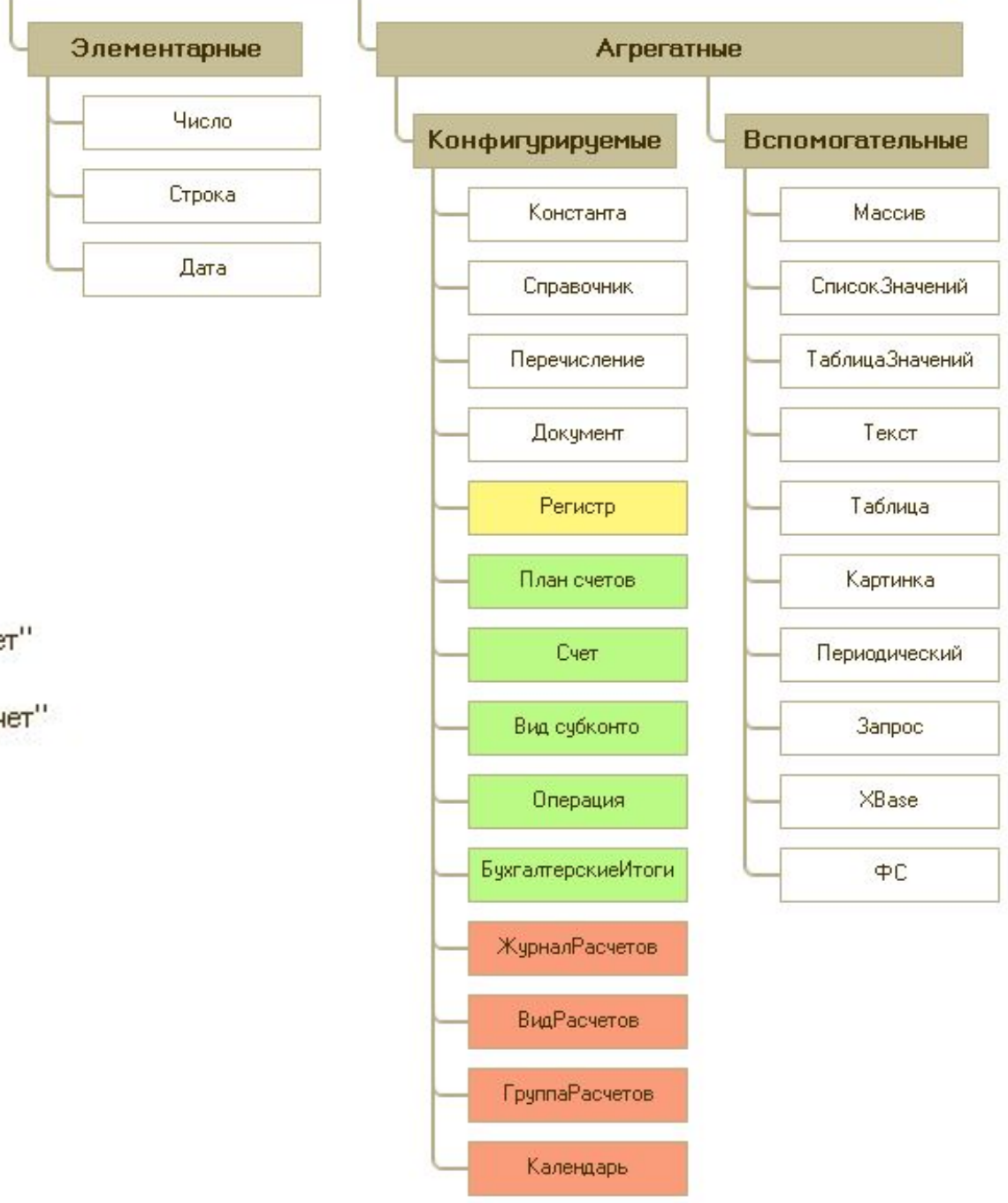
- **Конфигурируемые типы**, для которых возможна настройка производных объектов конфигурации в режиме 1С:Конфигуратор;
- **Вспомогательные типы**, для которых такая возможность отсутствует.

Типы данных системы

Кроме того, с точки зрения хранения информации, агрегатные типы данных делятся на:

- **Перманентные типы**, для которых создается структура хранения информации в базе данных или во внешних файлах;
- **Динамические типы**, значения которых хранятся в оперативной памяти и доступны лишь в текущем контексте выполнения;

Типы данных системы



Компонента "Оперативный учет"

Компонента "Бухгалтерский учет"

Компонента "Расчет"

Типы данных системы

К базовым типам данных в 1С относятся:

- числовой. В данных числового типа разрешается хранить любое десятичное число. Над ним определены арифметические и логические операции, операция присваивания. Явных ограничений на диапазон возможных значений переменных числового типа не накладывается;
- строковый. В переменных строкового типа имеется возможность представления произвольной последовательности символов (возможна и пустая строка), применимы операции объединения, логического сравнения, присваивания;
- дата. В переменных типа «дата» можно сохранять календарные даты в формате ДД.ММ.ГГ или ДД.ММ.ГГГГ (здесь ДД – это две цифры для представления дня месяца, ММ – номер месяца, ГГ (ГГГГ) – последние цифры (или полный номер) года).

Переменные в 1С: Предприятие

Переменная - это поименованная величина, которая может изменяться, принимая в процессе этого различные значения. В 1с применяется **динамическая типизация**, т.е. тип переменной определяется в момент присваивания значения, а не в момент объявления. Платформа поддерживает следующие типы переменных:

Тип	Описание
Строка	Любой текст
Число	Целые и вещественные числа
Булево	Логические значения Да и Нет
Дата	Дата и время
Объект	Множество различных типов

Переменные в 1С: Предприятие

Переменные могут создаваться с помощью зарезервированного слова Перем. Но внутри кода это необязательно. Можно просто присвоить переменной какое-либо значение, и если переменной с таким именем не было, она будет создана. Ключевое слово Перем обязательно использовать в 3 случаях:

1. если переменная впервые встречается справа от знака присваивания;
2. если переменная впервые встречается в параметре процедуры или функции;
3. нужно создать внешнюю переменную.

Присвоение переменной значения осуществляется с помощью оператора =:

```
Имя_Переменной = Новое_Значение;
```

ПРОСТЫЕ ЛОГИЧЕСКИЕ ВЫРАЖЕНИЯ В ЯЗЫКЕ 1С

К уже изученным типам данных (строка, число и дата) добавим еще один — логический тип. Он может принимать всего два значения: *Истина* или *Ложь.*

Значение логического типа (Истина или Ложь) является результатом вычисления некоторого логического выражения.
Логическое выражение можно определить как совокупность операндов, связанных между собой знаками логических операций.
Под операндами понимаются любые объекты, над которыми выполняются те или иные действия. В логических выражениях в качестве операндов могут использоваться отношения, представляющие собой выражение, составленное при помощи операций сравнения. Сравнивать можно числа, даты, строки и другие данные. О логическом выражении можно сказать, верно оно (Истина) или неверно {Ложь).

ПРОСТЫЕ ЛОГИЧЕСКИЕ ВЫРАЖЕНИЯ В ЯЗЫКЕ 1С

Операции сравнения бывают следующие:

Операция	Обозначение в языке 1С
Равно	=
Не равно	<>
Больше	>
Меньше	<
Больше или равно	>=
Меньше или равно	<=

ПРИМЕР

- 1 = 1 ("один равен одному").
- 4 <> 5 ("четыре не равно пяти")
- 3 > 1 ("три больше одного").

Обращаю ваше внимание, что перечисленные три примера логических выражений принимают значение *Истина*, так как все они верны.

ПРОСТЫЕ ЛОГИЧЕСКИЕ ВЫРАЖЕНИЯ В ЯЗЫКЕ 1С

ПРИМЕР

Пример логического выражения	Результат вычисления	Объяснение
1 =2	Ложь	неверно, единица на самом деле не равна двойке
1 = 1	Истина	верно, единица равна единице
"Земля" О "Луна"	Истина	верно,строка"Земля" не равна строке "Луна"
' 18610101' <>'18610101'	Ложь	неверно, дата 01.01.1861 на самом деле равна дате 01.01.1861
100 >50	Истина	верно, сто действительно больше пятидесяти
10 < 0	Ложь	неверно, на самом деле десять больше нуля

СЛОЖНЫЕ ЛОГИЧЕСКИЕ ВЫРАЖЕНИЯ В ЯЗЫКЕ 1С

Из простых логических выражений можно составлять сложные
логические выражения. Для этого существуют специальные логические операции: ***И***, ***Или***, ***Не***.

Правила работы логической

А (левая часть)	**Б (правая часть)**	**А И Б**	**Пример**
Истина	Истина	Истина	**(2 < 3) И (10 = 10)**
Истина	Ложь	Ложь	**(2 < 3) И (10 <> 10)**
Ложь	Истина	Ложь	**(10 <> 10) И (2 < 3)**
Ложь	**Ложь**	**Ложь**	**(1 = 2) И (3 = 5)**

СЛОЖНЫЕ ЛОГИЧЕСКИЕ ВЫРАЖЕНИЯ В ЯЗЫКЕ 1С

Таким образом, логическое выражение, составленное при помощи операции И, верно тогда и только тогда, когда верны оба выражения, стоящие слева и справа от этой операции.

Для лучшего понимания представьте, что ***значение Истина это 1, Ложь это 0, а логическая операция И это умножение:***

А (левая часть)	Б (правая часть)	А И Б
1	1	1
1	0	0
0	1	0
0	0	0

Правила работы логической операции Или

А (левая часть)	Б (правая часть)	А Или Б	Пример
Истина	Истина	Истина	(2 < 3) Или (10 = 10)
Истина	Ложь	Истина	(2 < 3) Или (10 <> 10)
Ложь	Истина	Истина	(10 <> 10) Или (2 < 3)
Ложь	Ложь	Ложь	(1 = 2) Или (3 = 5)

Таким образом, логическое выражение, составленное при помощи операции Или, истинно, если верно хотя бы одно из выражений, стоящих слева и справа от этой операции.

СЛОЖНЫЕ ЛОГИЧЕСКИЕ ВЫРАЖЕНИЯ В ЯЗЫКЕ 1С

Для лучшего понимания представьте, что значение Истина это 1, Ложь это 0, а логическая операция Или это сложение:

А (левая часть)	Б (правая часть)	А Или Б
1	1	1
1	0	1
0	1	1
0	0	0

Правила работы логической операции Не

А	Не А	Пример
Истина	Ложь	Не (2 < 3)
Ложь	Истина	Не (10 = 5)

Таким образом, логическое выражение, составленное при помощи операции *Не*, верно тогда и только тогда, когда неверно выражение, стоящее справа от этой операции. Операцию *Не* еще называют *логическим отрицанием*.

СЛОЖНЫЕ ЛОГИЧЕСКИЕ ВЫРАЖЕНИЯ В ЯЗЫКЕ 1С

Для лучшего понимания представьте, что значение Истина это 1, а Ложь это 0:

A	Не A
1	0
0	1

УСЛОВНАЯ КОМАНДА В ЯЗЫКЕ 1С

Частью условной команды являются следующие три слова:

Если, Тогда и КонецЕсли.

Оператор *Если* управляет выполнением программы, основываясь на результате одного или более логических выражений. Оператор может содержать любое количество групп операторов, возглавляемых конструкциями *ИначеЕсли — Тогда*.

```
Если <Логическое выражение> Тогда
        // Операторы
[ИначеЕсли <Логическое выражение> Тогда]
        // Операторы
[Иначе]
        // Операторы
КонецЕсли;
```

УСЛОВНАЯ КОМАНДА В ЯЗЫКЕ 1С

Параметры:

Если	Ключевое слово, которое начинает структуру оператора условного выполнения.
<Логическое выражение>	*Логическое выражение*.
Тогда	Операторы, следующие за *Тогда* выполняются, если результатом логического выражения является значение *Истина*.
// Операторы	Исполняемый оператор или последовательность таких операторов.
ИначеЕсли	Логическое выражение, следующее за ключевым словом *ИначеЕсли*, вычисляется только тогда, когда условия в *Если* и всех предшествующих *ИначеЕсли* оказались равны *Ложь*. Операторы, следующие за конструкцией *ИначеЕсли — Тогда*, выполняются, если результат логического выражения в данном *ИначеЕсли* равен *Истина*.
Иначе	Операторы, следующие за ключевым словом *Иначе*, выполняются, если результаты логических выражений в конструкции *Если* и всех предшествующих конструкциях *ИначеЕсли* оказались равны *Ложь*.
КонецЕсли	Ключевое слово, которое завершает структуру оператора условного выполнения.

УСЛОВНАЯ КОМАНДА В ЯЗЫКЕ 1С

Тернарный условный оператор

Позволяет вычислить одно из двух заданных выражений в зависимости от результата вычисления логического выражения.

```
?(<Логическое выражение>, <Выражение 1>, <Выражение 2>)
```

Параметры:

<Логическое выражение>	Логическое выражение, результат вычисления которого определяет одно из результирующих выражений, которые будут вычислены. Если результат его вычисления *Истина*, то будет вычисляться *<Выражение 1>*. Если результат *Ложь* – то *<Выражение 2>*.
<Выражение 1>	Результирующее выражение, которое будет вычисляться, если результат логического выражения *Истина*.
<Выражение 2>	Результирующее выражение, которое будет вычисляться, если результат логического выражения *Ложь*.

УСЛОВНАЯ КОМАНДА В ЯЗЫКЕ 1С

Условный оператор ЕСЛИ

```
Если 1 > 0 Тогда // Истина
// блок операторов
   Сообщить ("Компьютер выполнит все команды из этого блока." );
   Сообщить ("Один больше нуля." );
КонецЕсли;

Если 1 < 0 Тогда // Ложь
   Сообщить ("Один меньше нуля." );
Иначе
   Сообщить ("Сработает именно эта ветка условного оператора
(#А)." );
   Сообщить ("Один больше нуля." );
КонецЕсли;

Если 1 < 0 Тогда // Ложь
   Сообщить ("Один меньше нуля." );
ИначеЕсли 1 = 0 Тогда // Ложь
   Сообщить ("Один равен нулю." );
Иначе
   Сообщить ("Сработает именно эта ветка условного оператора
(#Б)." );
   Сообщить ("Один больше нуля." );
КонецЕсли;
```

Тернарный оператор ?

```
Текст = ?(1 > 2, "Один больше двух." , "Один не больше двух." );
Сообщить (Текст); // выведет "Один не больше двух."
```

ЦИКЛ "ДЛЯ" В ЯЗЫКЕ 1С

Цикл — это специальная команда компьютеру, которая позволяет повторять выполнение других команд нужное количество раз.

Выполнение цикла можно прервать в любой момент при помощи оператора **Прервать;** после него управление передастся операторам после КонецЦикла;. Так же можно прервать только текущую итерацию и перейти к следующей итерации при помощи оператор **Продолжить.**

Оператор цикла Для предназначен для циклического повторения операторов, находящихся внутри конструкции Цикл – КонецЦикла.
Перед началом выполнения цикла значение Выражение 1 присваивается переменной Имя_переменной.
Значение Имя_переменной автоматически увеличивается при каждом проходе цикла. Величина приращения счетчика при каждом выполнении цикла равна 1.
Цикл выполняется, пока значение переменной Имя_переменной меньше или равно значению Выражение 2. Условие выполнения цикла всегда проверяется в начале, перед выполнением цикла.

ЦИКЛ "ДЛЯ" В ЯЗЫКЕ 1С

Цикл — это специальная команда компьютеру, которая позволяет повторять выполнение других команд нужное количество раз.

```
Для <Имя_переменной> = <Выражение 1> По <Выражение 2> Цикл
    // Операторы
   [Прервать;]
   // Операторы
   [Продолжить;]
   // Операторы
   КонецЦикла;
```

ЦИКЛ "ДЛЯ" В ЯЗЫКЕ 1С

Параметры:

Имя_переменной	Идентификатор переменной (счетчика цикла), значение которой автоматически увеличивается на 1 при каждом повторении цикла. Так называемый счетчик цикла.
Выражение 1	Числовое выражение, которое задает начальное значение, присваиваемое счетчику цикла при первом проходе цикла.
По	Синтаксическая связка для параметра Выражение 2.
Выражение 2	Максимальное значение счетчика цикла. Когда переменная Имя_переменной становится больше чем Выражение 2, выполнение оператора цикла Для прекращается.
Цикл	Операторы, следующие за ключевым словом Цикл выполняются, пока значение переменной Имя_переменной меньше или равно значения Выражение 2.
// Операторы	Исполняемый оператор или последовательность таких операторов.
Прервать	Позволяет прервать выполнение цикла в любой точке. После выполнение этого оператора управление передается оператору, следующему за ключевым словом КонецЦикла.
Продолжить	Немедленно передает управление в начало цикла, где производится вычисление и проверка условий выполнения цикла. Операторы, следующие в теле цикла за ним, на данной итерации обхода не выполняются.
КонецЦикла	Ключевое слово, которое завершает структуру оператора цикла.

ЦИКЛ "ДЛЯ Каждого" В ЯЗЫКЕ 1С

Команды, заключенные между словами Цикл и Конец Цикла называются телом цикла и выполняются столько раз сколько нужно шагов, чтобы Начальное Число стало больше Конечного Числа.

Оператор цикла Для каждого предназначен для циклического обхода коллекций значений.

При каждой итерации цикла возвращается новый элемент коллекции.

Обход осуществляется до тех пор, пока не будут перебраны все элементы коллекции.

```
Для Каждого <Имя_переменной_1> Из <Имя_переменной_2> Цикл
// Операторы
[Прервать;]
// Операторы
 [Продолжить;]
// Операторы
КонецЦикла;
```

ЦИКЛ "ДЛЯ Каждого" В ЯЗЫКЕ 1С

Параметры:

Имя_переменной_1	Переменная, которой при каждом повторении цикла присваивается значение очередного элемента коллекции.
Из	Синтаксическая связка для параметра Имя_переменной_2.
Имя_переменной_2	Переменная или выражение, предоставляющее коллекцию. Элементы этой коллекции будут присваиваться параметру Имя_переменной_1.
Цикл	Операторы, следующие за ключевым словом Цикл выполняются для каждого элемента коллекции.
// Операторы	Исполняемый оператор или последовательность таких операторов.
Прервать	Позволяет прервать выполнение цикла в любой точке. После выполнение этого оператора управление передается оператору, следующему за ключевым словом КонецЦикла.
Продолжить	Немедленно передает управление в начало цикла, где производится вычисление и проверка условий выполнения цикла. Операторы, следующие в теле цикла за ним, на данной итерации обхода не выполняются.
КонецЦикла	Ключевое слово, которое завершает структуру оператора цикла.

ЦИКЛ «Пока" В ЯЗЫКЕ 1С

Оператор цикла Пока предназначен для циклического повторения операторов, находящиеся внутри конструкции Цикл – КонецЦикла. Цикл выполняется, пока логическое выражение равно Истина. Условие выполнения цикла всегда проверяется вначале, перед выполнением цикла.

```
Пока <Логическое выражение> Цикл
// Операторы
[Прервать;]
// Операторы
[Продолжить;]
// Операторы
КонецЦикла;
```

ЦИКЛ «Пока" В ЯЗЫКЕ 1С

Параметры:

Логическое выражение	Логическое выражение.
Цикл	Операторы, следующие за ключевым словом Цикл, выполняются, пока результат логического выражения равен *Истина*.
// Операторы	Исполняемый оператор или последовательность таких операторов.
Прервать	Позволяет прервать выполнение цикла в любой точке. После выполнение этого оператора управление передается оператору, следующему за ключевым словом КонецЦикла.
Продолжить	Немедленно передает управление в начало цикла, где производится вычисление и проверка условий выполнения цикла. Операторы, следующие в теле цикла за ним, на данной итерации обхода не выполняются.
КонецЦикла	Ключевое слово, которое завершает структуру оператора цикла.

Комбинирование простых конструкций В ЯЗЫКЕ 1С

Пусть у нас есть два цикла:

```
Для Шаг1 = 1 По 2 Цикл
    Сообщить(Шаг1);
КонецЦикла;
```

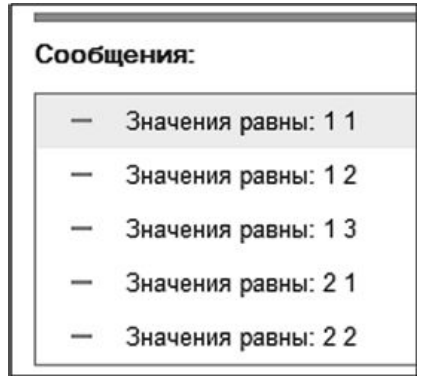
```
Для Шаг2 = 1 По 3 Цикл
    Сообщить(Шаг2);
КонецЦикла;
```

Эти циклы хорошо нам знакомы и мы понимаем, что в результате первого цикла выведутся числа от 1 до 2, а в результате второго — от 1 до 3.

Если мы вложим выполнение одного цикла внутрь другого? Вот так:

```
Для Шаг1 = 1 По 2
 Цикл
Для Шаг2 = 1 По 3Цикл
  Сообщить("Значения равны: " +
 Шаг1 + " " + Шаг2);
  КонецЦикла;
КонецЦикла;
```

Если запустить этот пример на компьютере, то получатся следующие результаты:



Массивы в ЯЗЫКЕ 1С

Массив — это программная коллекция (объект встроенного языка), содержащая пронумерованную последовательность значений произвольного типа. Каждому элементу присваивается последовательный целочисленный номер — **индекс**, начинающийся с нуля. По индексу можно получить или установить значение элемента массива.

Создание пустого массива выполняется с помощью оператора Новый

```
Массив = Новый Массив;
```

Создание добавления элементов в массив используется

```
Массив.Добавить();
```

Каждый новый элемент помещается в конец массива. Чтобы создать массив с заданным размером, можно воспользоваться конструктором массива. Например, так:

```
Массив = Новый Массив(3);
```

В 1С:Предприятие 8 существует два типа массивов:

- **фиксированный массив** — такой массив имеет фиксированный размер, заданный при его создании. Для фиксированного массива недоступно программное изменение размера, количества и последовательности элементов;
- **обычный массив** — такой массив имеет динамический размер и его верхний предел практически неограничен. Таким массивом можно произвольно оперировать из встроенного языка;

Массивы в ЯЗЫКЕ 1С

Для доступа к отдельным элементам массива применяется **операция разыменования**: указывается имя переменной, объявленной как массив, а затем, в квадратных скобках, указывается индекс элемента:

```
ИмяМассива[Индекс]
```

Такие конструкции можно указывать как слева от оператора присваивания:

```
Массив[0] = 10; Массив[1] = Дата(2018, 1, 1); Массив[2] = "Просто строка";
```

Так и справа: в выражениях и в качестве параметров процедур и функций:

```
Сообщить("Массив[0] = " + Массив[0]);
```

Левая граница массива постоянна и всегда равно нулю (0). Верхняя граница напрямую связана с количеством элементов в массиве. Получить ее можно двумя способами:

- С использованием метода Массив.Количество() с последующим вычитанием 1;
- С использованием одноименного метода Массив.ВГраница();

Массивы в ЯЗЫКЕ 1С

Поиск в массиве:
Для поиска значений в массиве можно воспользоваться одноименным методом Массив.Найти (). Он возвращает индекс найденного элемента или Неопределено, если таковой не был найден:

```
НайдЭлт = Массив.Найти(10);
Если НЕ НайдЭлт = Неопределено Тогда
Сообщить("Массив[" + НайдЭлт + "] = " + Массив[НайдЭлт])
КонецЕсли;
```

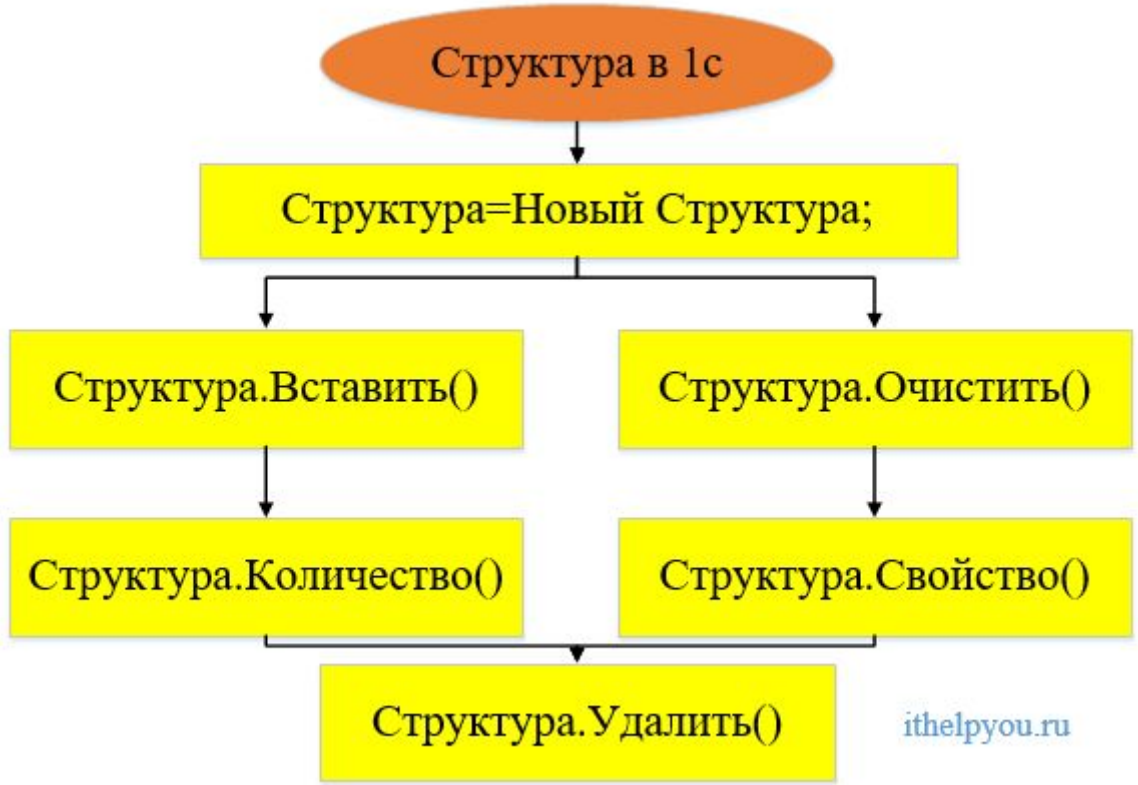
Методы объекта Массив

ВГраница()	Возвращает старший индекс массива
Вставить()	Добавляет значение в произвольное место массива
Добавить()	Добавляет значение в конец массива
Количество()	Возвращает количество элементов в массиве
Найти()	Возвращает индекс значения в массиве
Очистить()	Удаляет все элементы массива
Получить()	Возвращает значение элемента массива по его индексу
Удалить()	Удаляет значение из массива по его индексу
Установить()	Устанавливает значение элемента массива

Структура в ЯЗЫКЕ 1С

Структура в 1с 8.3- это программная **коллекция значений** , содержащая набор элементов “Ключ” и “Значение”. То есть , Ключ структуры является строковым идентификатором, по которому можно получать или устанавливать Значение. Идентификатором может выступать только строковое значение, т. е. значение типа строка.

Каждый элемент структуры является программным объектом “Ключ” и “Значение” встроенного языка, он содержит следующие свойства:
Ключ(тип Строка)-строковый идентификатор;
Значение(Произвольный тип)-произвольное значение.



Структура в ЯЗЫКЕ 1С

Методы объекта Структура

Метод	Описание
Вставить()	Добавляет новое свойство в Структуру
Количество()	Возвращает количество элементов в Структуре
Очистить()	Удаляет все элементы Структуры
Свойство()	Реализует безопасное чтение значения свойства
Удалить()	Удаляет свойство из Структуры по имени

Добавление элементов в структуру 1с

Для добавления новых элементов Структуры и замещения существующих используется метод Структура.Вставить():

```
Структура.Вставить("Код", 101); Структура.Встав
("Наименование", "Стол обеденный");
```

Обращение к свойствам структуры 1С

Для доступа к значениям Структуры применяется операция разыменования: указывается имя переменной, объявленной как Структура, а затем Ключ значения через точку. Разыменование возможно как слева от оператора присваивания, так и справа:

```
Структура.Код="200"; Структура.Наименование=Структура.Код;
```

Перебор элементов структуры 1С

Для перебора всех элементов структуры используется оператор цикла Для Каждого:

```
Для Каждого Элемента из Структура Цмкл Сообщить(Элемент.
Ключ+"="+Элемент.Значение); Конеццикла;
```

Процедуры и Функции в ЯЗЫКЕ 1С

Процедура – это идентификатор, выполняющий некоторые действия, определенные пользователем или системные. Если мы пропишем внутри нашего кода некоторую процедуру (не важно, нами написанную или процедуру программы 1С), то когда выполнение программы дойдет до этой строки, тогда будут выполнены определенные действия.

```
Процедура ИмяПроцедуры() //оператор
КонецПроцедуры
```

Функция – это идентификатор, выполняющий некоторые действия, определенные пользователем или системные, и возвращающий значение определенного типа. Функции отличаются от процедур только тем, что возвращают определенное значение. Например, функция СтрДлина() вычисляет длину строки. Аргумент передается в нее в качестве параметра, и возвращает это значение пользователю в виде числа. Или функция ТекущаяДата() определяет системную дату компьютера и возвращает это значение в виде даты.

```
Функция ИмяФункции() //оператор Возврат
Результата; Конецфункции
```

Процедуры и Функции в ЯЗЫКЕ 1С

Процедуры и функции в 1с могут обмениваться параметрами или аргументами. Передача параметров процедур и функций выполняется одним из способов.

Во-первых способ называется передачей по ссылке и представляет собой передачу не конкретного значения параметра, а адреса памяти (ссылки на переменную), где расположено это значение. Изменение переданного значения в вызываемой процедуре или функции приведет к изменению передаваемой переменной в вызывающем методе.
Во-вторых способ называется передачей по значению и представляет собой передачу копии значения параметра. В этом случае изменение переданного значения в вызываемой процедуре или функции не приведет к изменению значения передаваемой переменной в вызывающем методе.
В то же время механизм передачи параметров процедур и функций зависит от того, какой вызов будет выполняться:

- вызов без передачи управления между клиентом и сервером (только на клиенте или только на сервере);
- вызов с передачей управления между клиентом и сервером.

Отладчик в ЯЗЫКЕ 1С

Отладчик является встроенным в конфигуратор инструментом. Он помогает отлаживать программные модули, создаваемые в процессе разработки прикладного решения. Отладчик позволяет отслеживать последовательность выполнения операторов встроенного языка и просматривать значения переменных.

Основные возможности отладки

- отладка приложений, исполняемых на удаленных компьютерах, доступных по протоколу TCP/IP или HTTP,
- отладка кода, исполняемого рабочим процессом кластера серверов 1С:Предприятия 8; при этом поддерживается сквозной стек вызовов для клиента и сервера, и сквозная пошаговая отладка клиента и сервера;
- отладка кода, исполняемого в таких видах соединений как внешнее соединение, фоновое задание и WS-соединение;
- отладка мобильных приложений.

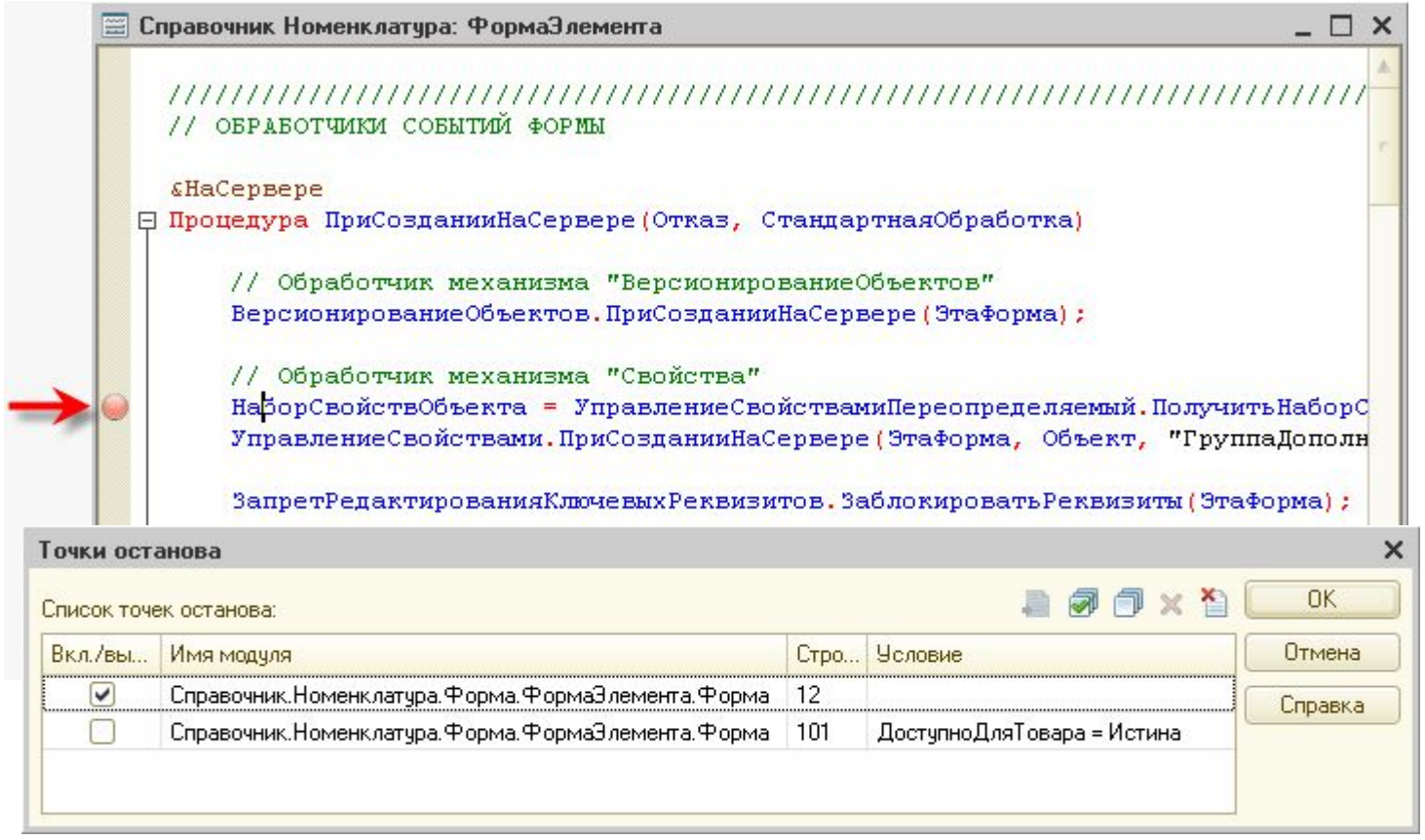
Отладчик в ЯЗЫКЕ 1С

Точки останова

Отладчик позволяет установить на конкретную строку модуля специальный маркер — точку останова, — при достижении которой исполнение программного модуля останавливается и управление передается отладчику. Точки останова могут быть безусловными или с условием. При достижении безусловной точки останова исполнение программного модуля останавливается в любом случае:

При достижении точки останова с условием, выполнение программного модуля останавливается только в том случае, если заданное условие истинно. Отладчик поддерживает возможность отключения точек останова. При этом строка модуля остается отмечена маркером, однако на ход исполнения модуля он никакого влияния не оказывает.

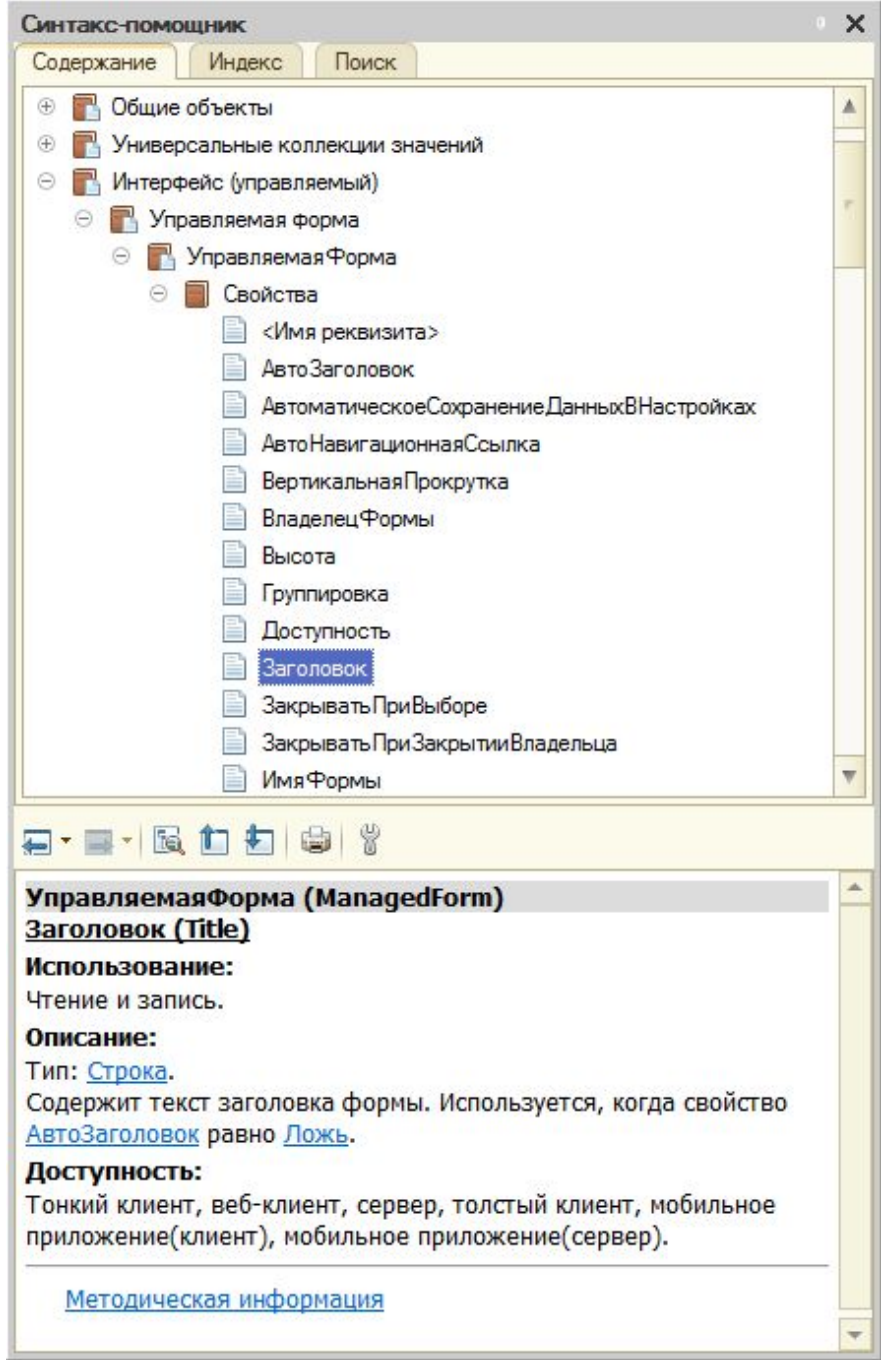
При большом количестве точек останова удобно использовать отдельное окно для работы с точками останова, позволяющее просматривать и редактировать их в едином списке.



Синтаксис помощник в ЯЗЫКЕ 1С

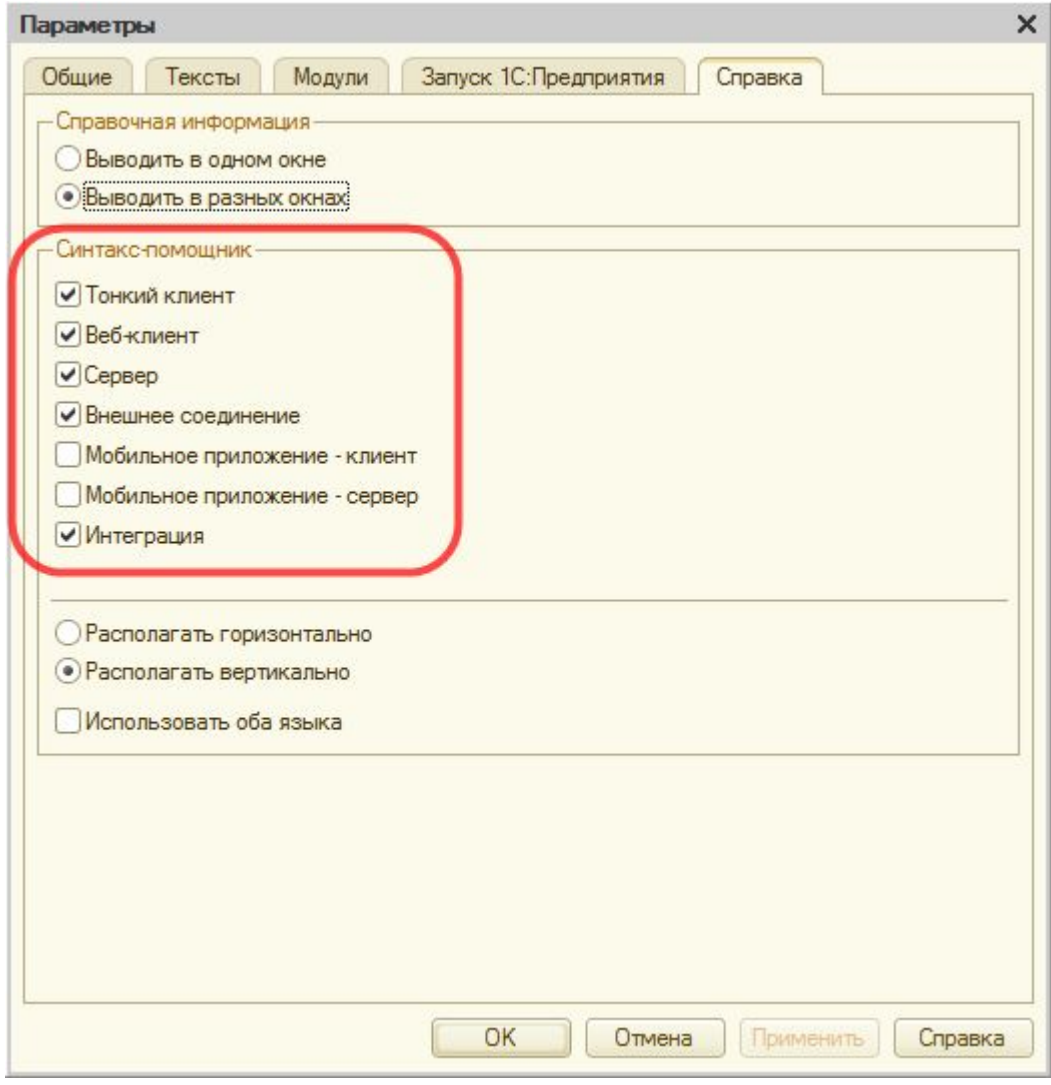
Синтакс-помощник — один из инструментов разработки. Он позволяет быстро получить подсказку по синтаксису встроенного языка в процессе написания кода программы.

Синтакс-помощник реализован в составе конфигуратора и содержит информацию об объектах встроенного языка, их свойствах, методах и связанных с ними событиях. В верхней части окна синтакс-помощника в виде дерева отображается список элементов встроенного языка: операторов, управляющих конструкций, процедур и функций, системных констант и др. Для удобства все элементы встроенного языка объединены в тематические разделы, представленные в виде ветвей дерева.



Синтаксис помощник в ЯЗЫКЕ 1С

В нижней части окна синтакс-помощника для каждого свойства, метода или события отображается подробная информация о синтаксисе, параметрах и особенностях использования элемента встроенного языка, выбранного в верхнем окне. В синтакс-помощнике поддерживаются гиперссылки на упоминаемые объекты встроенного языка, что позволяет быстро переходить к интересующей информации.



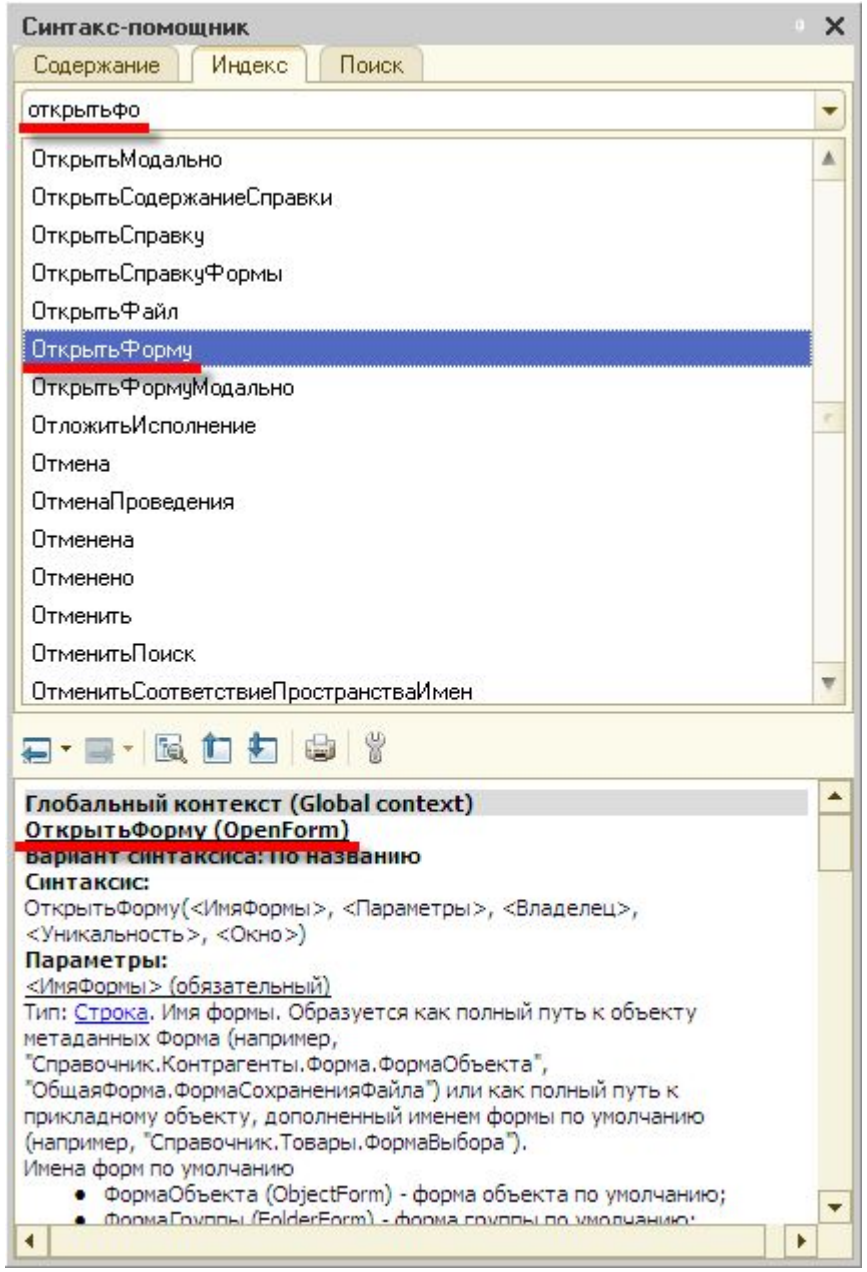
Отбор по контексту

Синтакс-помощник может отображать всю информацию или только ее часть, относящуюся к выбранным контекстам исполнения:

Синтаксис помощник в ЯЗЫКЕ 1С

Поиск по началу фразы

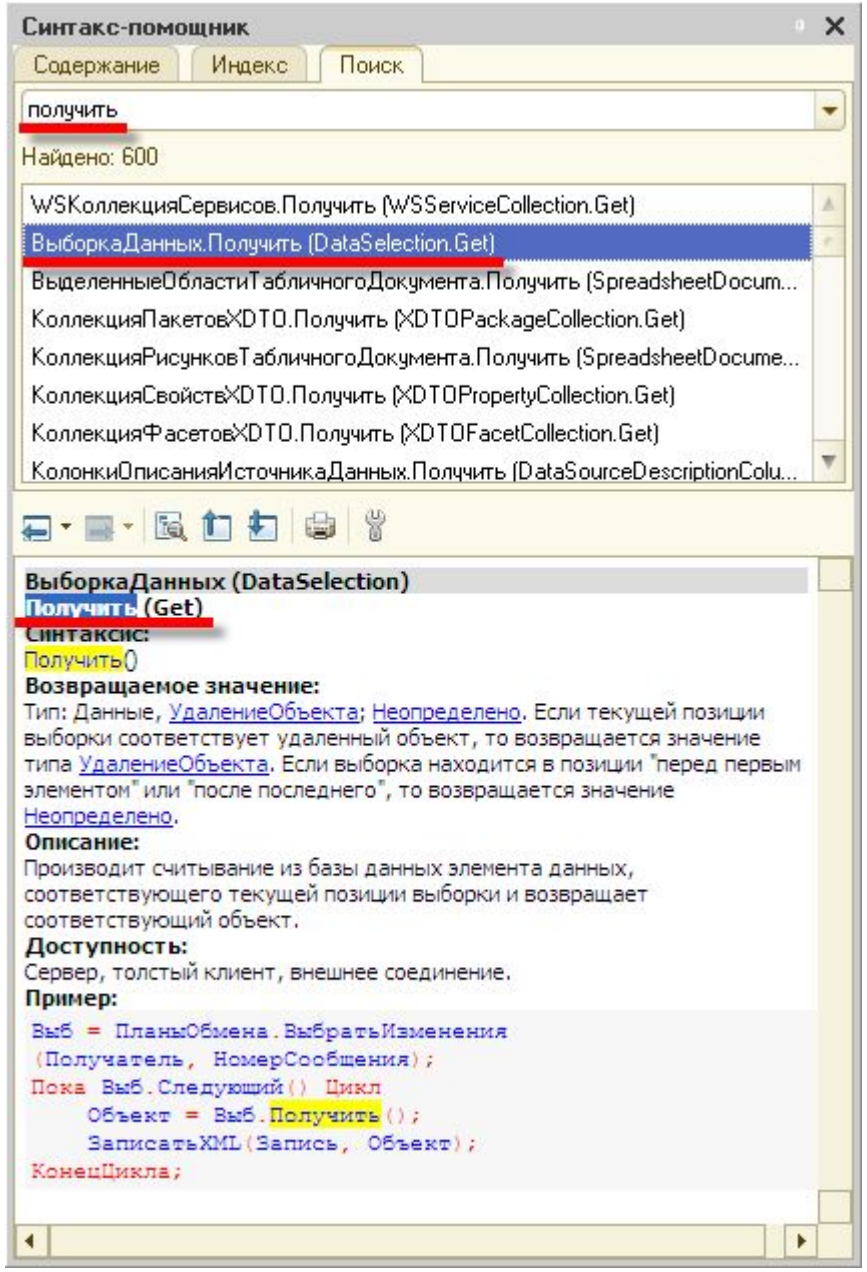
Синтакс-помощник позволяет осуществлять поиск по строке с последующим выбором конкретного объекта встроенного языка, описание которого интересует.



Синтаксис помощник в ЯЗЫКЕ 1С

Полнотекстовый поиск

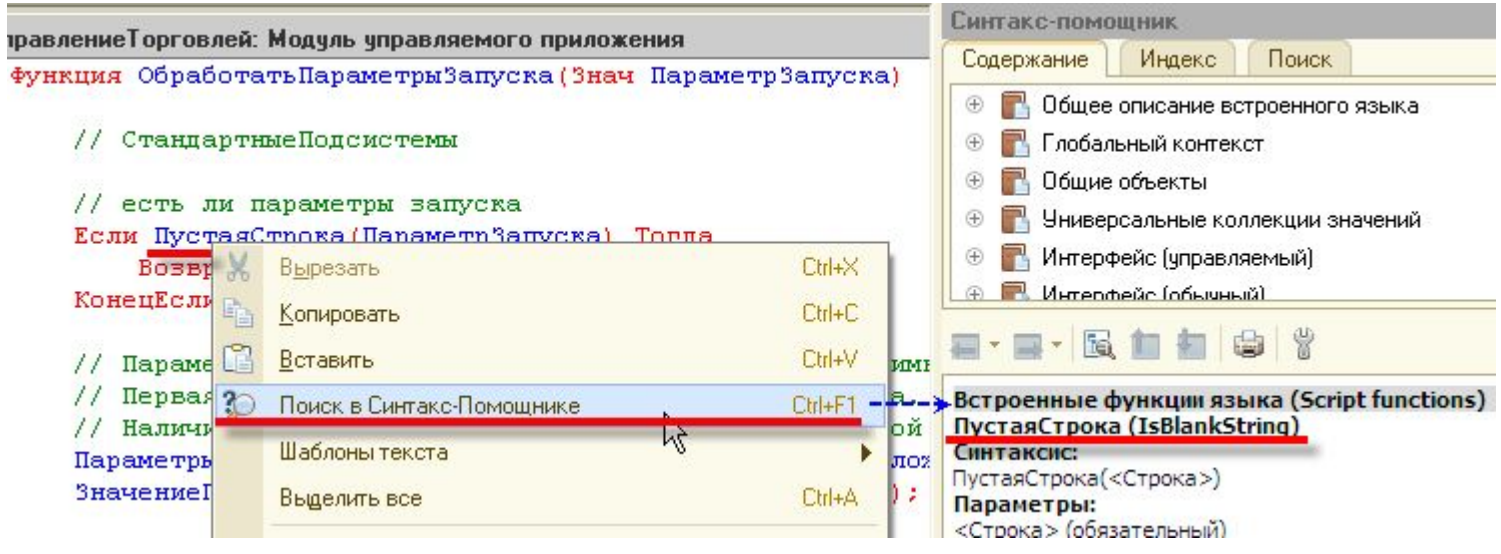
Также имеется возможность полнотекстового поиска по синтакс-помощнику. Найденные слова автоматически выделяются в тексте.



Синтаксис помощник в ЯЗЫКЕ 1С

Быстрое получение справки при редактировании текста программы

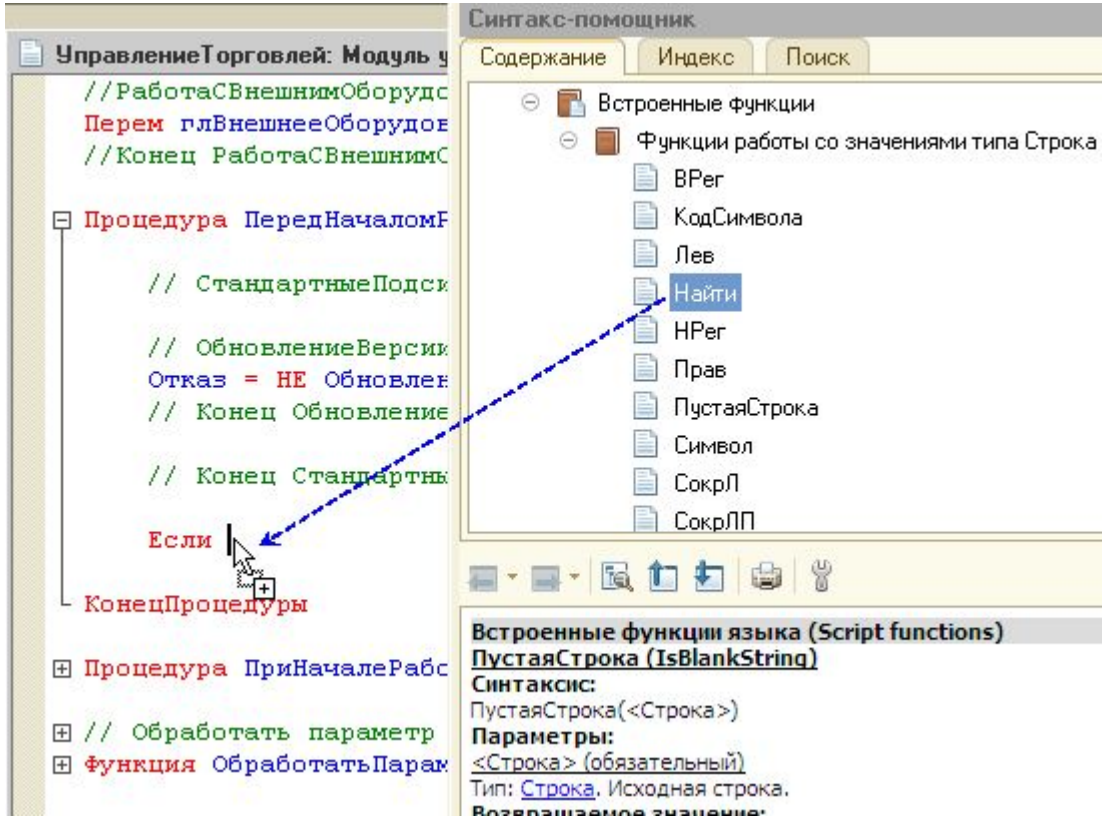
Открыв в текстовом редакторе программный модуль, можно установить курсор на интересующую конструкцию встроенного языка и по контекстному меню, или по горячей клавише, сразу перейти к описанию этой конструкции в синтакс-помощнике.



Синтаксис помощник в ЯЗЫКЕ 1С

Размещение готовых конструкций встроенного языка в тексте программы

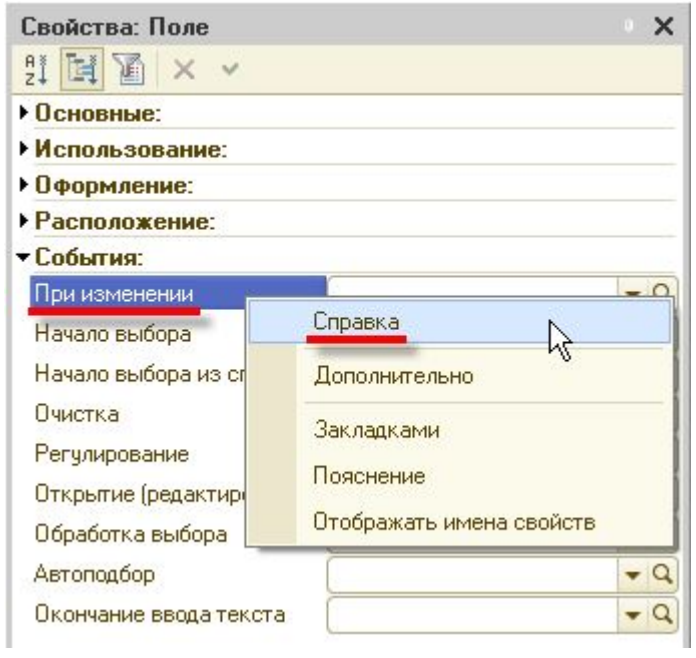
Готовые конструкции встроенного языка можно размещать в модуле просто перетащив их мышью из синтакс-помощника.



Синтаксис помощник в ЯЗЫКЕ 1С

Быстрое получение справки в палитре свойств

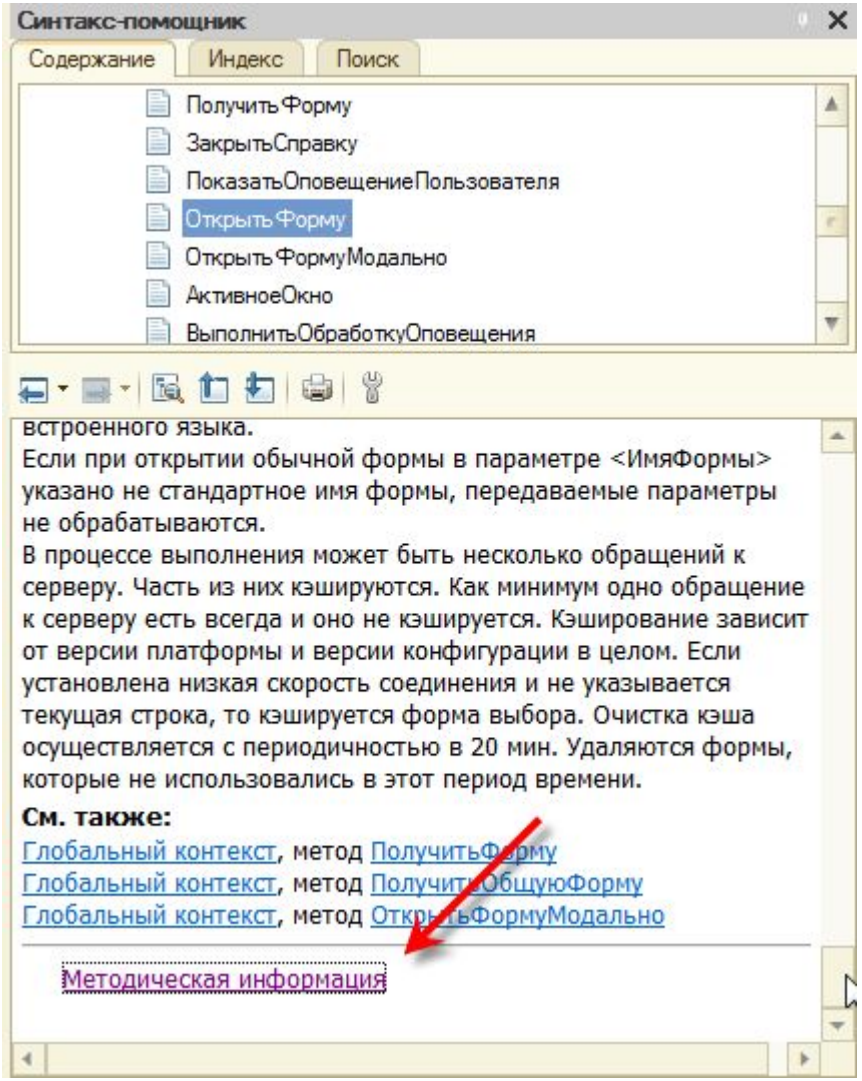
Находясь в палитре свойств можно быстро получить справку по выбранному методу, свойству или событию с помощью контекстного меню палитры свойств.



Синтаксис помощник в ЯЗЫКЕ 1С

Методические ссылки

В синтакс-помощнике, в конце каждого описания появилась ссылка Методическая информация:



Дата в ЯЗЫКЕ 1С

Дата — это один из примитивных типов данных существующих в 1С.
Значение типа дата в 1С содержит точную (с точностью до 0,1 миллисекунды) дату григорианского календаря.
Типовой порядок частей даты принятый в 1С — год, месяц, день, час, минута, секунда.
создания новой переменной типа дата является приравнивание к переменной строки цифр в одинарных кавычках (обязательно) вида ‘ГГГГММДДччммсс’ где:
ГГГГ — четыре цифры года (включая тысячелетие и век);
ММ — две цифры месяца;
ДД — две цифры даты;
чч — две цифры часа (в 24-х часовом формате);
мм — две цифры минут;
сс — две цифры секунд.
Часы, минуты и секунды можно не указывать, при этом они приравниваются к нулю.

Функция Дата() служит для создания значения типа “Дата”. Использовать ее можно двумя способами:

- Дата(x) — где параметр “x” это строка вида “ГГГГММДДччммсс”, часы, минуты и секунды можно опустить, т.е. преобразуем строку в дату;
- Дата(x, y, z, w, v, c) — в этом случае параметры “x”, “y”, “z”, “w”, “v” и “c” являются числами и означают год, месяц, день, час, минуту и секунду соответственно. Как и в предыдущем случае час, минуту и секунду можно опустить.

Дата в ЯЗЫКЕ 1С

Функции вида “Начало…(х)” получают дату в параметре “х” и возвращают дату начала периода времени в соответствии с названием конкретной функции. Существуют следующие функции:

- НачалоГода;
- НачалоКвартала;
- НачалоМесяца;
- НачалоНедели;
- НачалоДня;
- НачалоЧаса;
- НачалоМинуты.

Функции вида “Конец…(х)” получают дату в параметре “х” и возвращают дату окончания периода времени в соответствии с названием конкретной функции. Существуют следующие функции:

- КонецГода;
- КонецКвартала;
- КонецМесяца;
- КонецНедели;
- КонецДня;
- КонецЧаса;
- КонецМинуты.

Формат в ЯЗЫКЕ 1С

Функция Формат в 1С 8.3 выводит примитивные типы данных (дата, время, число, булево) в удобное отображение для чтения. Используется при визуальном выводе информации на экран. Синтаксис функции: Формат(Значение - форматируемое значение, Форматная строка - строковое значение, включающее параметры форматирования).

Функция Формат()
Синтаксис функции достаточно прост:

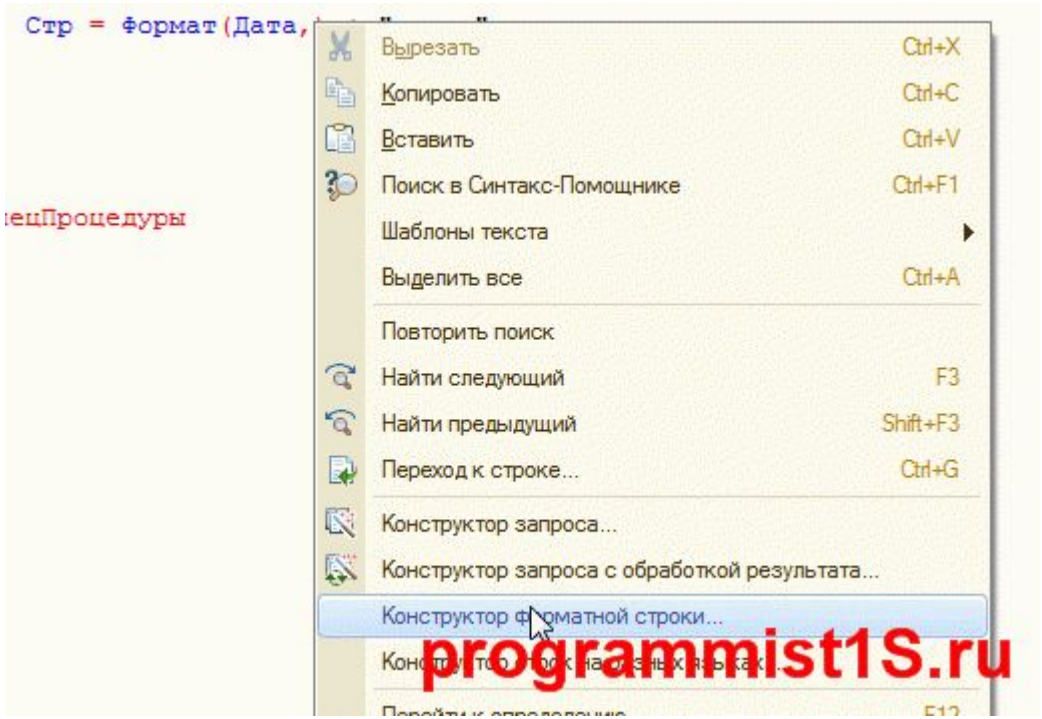
```
Формат(<Значение>,<Форматная строка>)
```

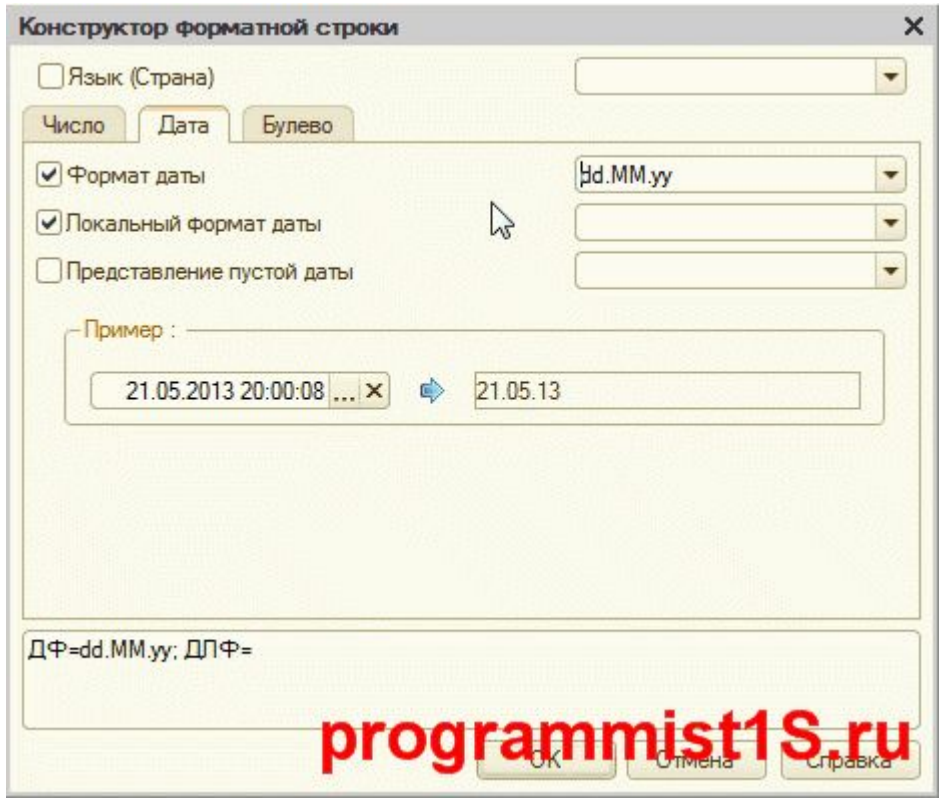
Значение — форматируемое значение, **Форматная строка** — строка, заданная определенным образом, из которой формируется правило обработки формата.

Формат в ЯЗЫКЕ 1С

Конструктор форматной строки

Для облегчения труда разработчика фирма 1С встроила в платформу специальный конструктор форматной строки.





Где необходимо выбрать нужную Вам вкладку в зависимости от типа данных — Число, Дата или Булево.

Использование диалогов в ЯЗЫКЕ 1С

Диалог в 1С 8.3 - это окно (элемент графического интерфейса), которое открывается пользователю (независимо от формы) для вывода информации/вопросов, получение ответов, ввода информации. Диалоги могут быть модальные и немодальные. Модальные диалоги блокируют работу с другими окнами формы до своего закрытия. Обычно с их помощью реализуется диалог, требующий от пользователя принятия некоторого решения. Соответственно немодальные диалоги могут параллельно отрываться и работать одновременно с другими окнами формы.

Вывод сообщений в пользовательском режиме решает **ряд задач:**
отражение хода выполнения текущего процесса (показ стадии выполнения процесса;
показ расчетных значений, полученных в ходе работы алгоритма);
выдача ошибок пользователю для возможного их исправления;
выдача рекомендаций.
Типы сообщений: терминирующие, которые останавливают выполнение программы и не дают продолжить ее, пока пользователь не ознакомится с этим сообщением и не выполнит определенные действия. Например, на экран пользователю будет выдан вопрос, на который нужно будет ответить Да или Нет. Пока пользователь не ответит – программа не выполняет дальнейшие действия;
ознакомительные сообщения, которые просто выводятся для пользователя и позволяют работать дальше (т.е. используются в режиме оповещения).

Использование диалогов в ЯЗЫКЕ 1С

Терминирующими сообщениями должны быть сообщения об ошибках, а ознакомительными: рекомендации, сообщения о текущем этапе процесса и показ расчетных значений (отладочная печать).

Ознакомительные сообщения предназначены для того, чтобы выдать пользователю некоторую информацию. Необходимо, чтобы пользователь с ней обязательно ознакомился и, возможно, предпринял какие-то действия, которые описаны в этом сообщении. Очень важно, чтобы пользователь действительно читал эти сообщения, поэтому они должны содержать только важную информацию. Тестовые и отладочные сообщения выдавать пользователю не стоит, т.к. рано или поздно он начнет игнорировать абсолютно все сообщения. В концепции управляемого интерфейса несколько изменился подход к выдаче сообщения. Оно теперь привязано к форме, в которой возникло. Его уже нельзя закрыть так, чтобы текст было совсем невидно.

Список значений в ЯЗЫКЕ 1С

Список значений — это универсальная коллекция для хранения некого списка значений. Список значений может хранить значения разных типов. ЭлементСпискаЗначений имеет 4 свойства:
Значение,
Представление,
Пометка,
Картинка.
Список значений можно использовать для интерактивных действий с пользователем, например выбор документа из списка. Можно использовать и на клиенте и на сервере, а также передавать с клиента на сервер. В обычном приложении нельзя передавать на сервер.

Список значений 1С с натяжкой можно назвать расширенным аналогом массива 1С. Это также – набор значений (значения могут разных типов), есть поиск и сортировка значений.

В списке значений отсутствует:
- список значений 1С создается пустой, а значений добавляются потом (т.е. недоступно: Массив = Новый Массив(10))
- многомерность (хотя элементом списка значений 1С может быть другой элемент массива значений)
- метод ВГраница()

Список значений в ЯЗЫКЕ 1С

В списке значений 1С есть дополнительные возможности:

- в массиве «элемент» — это непосредсвенное значение, а в списке значений 1С «элемент» это структура со свойствами .Значение и .Представление
- каждый элемент списка значений 1С отображается в интерфейсе не по значению, а по представлению (если оно задано), которое может сильно отличаться от значения (так как задал программист)
- у каждого элемента может быть отображена картинка и «чекбокс» (квадратик для установки галочки «отмечено или нет»)
- у списка значений 1С есть метод выбора пользователем значения из списка.

Список значений 1С часто используется:
А) Как обычный массив
Б) Для работа со списком значений 1С на интерфейсе (например выбор значения в «выпадающем списке»)
В) Для отборов в интерфейсных списках (отбор по множеству значений).
Список значений 1С можно использовать для отбора в запросе полностью также, как и массив.

Варианты работы системы

Платформа поддерживает два варианта работы: файловый и клиент-серверный. И в том, и в другом варианте все прикладные решения работают полностью идентично.

Файловый вариант работы рассчитан на персональную работу одного пользователя или работу небольшого количества пользователей в локальной сети. В этом варианте все данные информационной базы располагаются в одном файле — в файловой СУБД.

Клиент-серверный вариант работы предназначен для использования в рабочих группах или в масштабе предприятия. Он реализован на основе трехуровневой архитектуры «клиент-сервер». В этом варианте информационная база хранится в одной из поддерживаемых систем управления базами данных, а взаимодействие между клиентским приложением и СУБД осуществляет кластер серверов «1С:Предприятия 8».

Работа под управлением различных операционных систем

Все основные компоненты платформы способны функционировать как под управлением операционной системы Windows, так и под управлением операционной системы Linux. Благодаря тому, что взаимодействие процессов между собой осуществляется по протоколу TCP/IP, в составе системы могут присутствовать компоненты с различными операционными системами.

Варианты работы системы

Кроссплатформенность - это способность системы работать под управлением различных операционных систем. Основные компоненты системы могут работать как под управлением операционной системы Windows, так и под управлением операционной системы Linux. Кроме этого клиентская часть «1С: Предприятия 8» может быть запущена и на компьютерах с операционными системами macOS и iOS.

Варианты работы системы

Подключение через Интернет позволяет обеспечить удаленную on-line работу пользователей с информационными базами. Это возможно благодаря использованию тонкого клиента, веб-клиента или мобильного клиента. Они подключаются к специальным образом настроенному веб-серверу, который осуществляет их взаимодействие с кластером или с файловой информационной базой.

Клиенты могут использовать различные способы выхода в Интернет. Это могут быть высокоскоростные подключения по выделенным линиям или через локальную сеть. А могут быть низкоскоростные подключения, например, через мобильное GPRS-соединение.

Варианты работы системы

Файловый вариант работы — один из вариантов работы системы «1С:Предприятие 8». Файловый вариант работы рассчитан на персональную работу одного пользователя или работу небольшого количества пользователей в локальной сети.
В этом варианте все данные информационной базы (конфигурация, база данных, административная информация) располагаются в одном файле — файловой базе данных. Работу с этой базой данных осуществляет файловая СУБД, разработанная фирмой «1С» и являющаяся частью платформы.

Такой вариант работы обеспечивает легкость установки и эксплуатации системы. При этом для работы с информационной базой не требуются дополнительные программные средства, достаточно иметь операционную систему и «1С:Предприятие 8».
Файловый вариант работы обеспечивает целостность информационной базы и простое создание резервных копий. Исключена ситуация, когда пользователь может по ошибке (например, при копировании информационной базы) перепутать различные файлы информационной базы и привести, таким образом, систему в неработоспособное состояние.

Варианты работы системы

Клиент-серверный вариант работы — один из вариантов работы системы «1С:Предприятие 8». Клиент-серверный вариант работы предназначен для использования в рабочих группах или в масштабе предприятия. Он реализован на основе трехуровневой архитектуры «клиент-сервер».

Клиент-серверная архитектура разделяет всю работающую систему на три различные части, определенным образом взаимодействующие между собой:
клиентское приложение,
кластер серверов «1С:Предприятия 8»,
сервер базы данных.

Варианты работы системы

Работа в клиент-серверном варианте возможна как напрямую с кластером, так и через веб-сервер. При этом в случае непосредственного подключения к кластеру толстый клиент и тонкий клиент используют протокол TCP/IP. При подключении через веб-сервер тонкий клиент и веб-клиент используют протокол HTTP или HTTPS.

Варианты работы системы

Кластер серверов «1С:Предприятия 8» — основной компонент платформы, обеспечивающий взаимодействие между пользователями и системой управления базами данных в клиент-серверном варианте работы. Наличие кластера позволяет обеспечить бесперебойную, отказоустойчивую, конкурентную работу большого количества пользователей с крупными информационными базами.

Кластер серверов «1С:Предприятия 8» является логическим понятием и представляет собой совокупность рабочих процессов, обслуживающих один и тот же набор информационных баз.

Сервер баз данных

В качестве сервера баз данных могут использоваться:

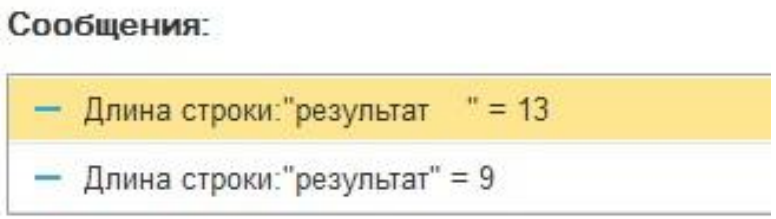
- Microsoft SQL Server,
- PostgreSQL,
- IBM DB2,
- Oracle Database.

Строки в ЯЗЫКЕ 1С

Строка — это один из примитивных типов данных существующих 1С и работать с ним приходится практически постоянно. Следовательно необходимо иметь представление о том, какие функции для работы со строками существуют, как и когда они используются, а также, что получается в результате.

ДЛИНА СТРОКИ 1С

В платформе 1С имеется функция, которая вычисляет длину строки – СтрДлина. Эта функция имеет единственный параметр – строку, и возвращает количество символов в этой строке, т.е. её длину. Причем считаются все символы, в том числе пробелы.



Строки в ЯЗЫКЕ 1С

РЕГИСТР СТРОКИ 1С

В 1С имеются функции для работы с регистрами строк.
НРег – переводит строку в нижний регистр
ВРег – переводит строку в верхний регистр
ТРег – переводит строку в титульный регистр (каждое слово начинается с заглавной буквы).

```
СтрокаПримера= "пРивет оЛег!"; Сообщить(НРег(СтрокаПримера));
Сообщить(ВРег(СтрокаПримера)); Сообщить(ТРег(СтрокаПримера));
```

УБРАТЬ ПРОБЕЛЫ В СТРОКЕ 1С

Часто возникает, что в строке 1С в начале строки или в конце строки есть лишние пробелы, которые нужно убрать. Для этих целей служат следующие функции.

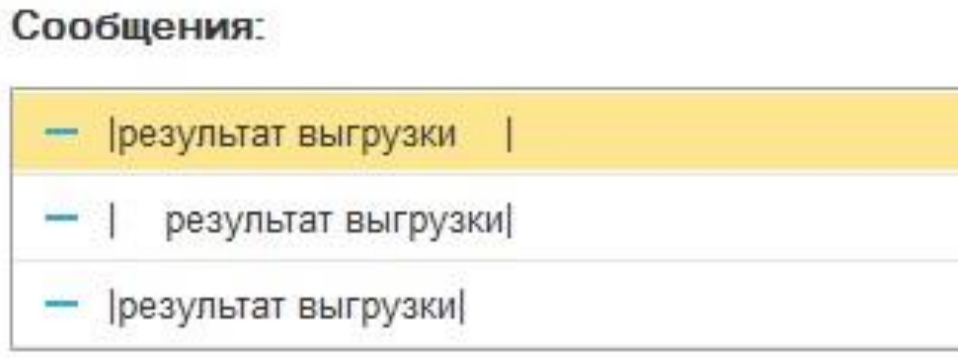
СокрЛ — убирает пробелы слева строки.

СокрП — убирает пробелы справа строки.

СокрЛП – убирает пробелы справа и слева строки.

Рассмотрим пример (добавлю символы перед и после функциями, чтобы было понятно как они работают).

```
СтрокаПримера= " результат выгрузки "; Сообщить("|" + СокрЛ
(СтрокаПримера) + "|"); Сообщить("|" + СокрП(СтрокаПримера) + "|");
Сообщить("|" + СокрЛП(СтрокаПримера) + "|");
```



Строки в ЯЗЫКЕ 1С

СОКРАТИТЬ СТРОКУ 1С

Если предыдущие функции сокращали только пробелы, то в платформе 1С имеется возможность сократить и саму строку на нужное количество символов. Или наоборот – оставить нужное количество символов. Для этих целей служат следующие функции.

Лев – оставляет нужное количество символов слева. Имеет следующий синтаксис: Лев (<Строка>,<ЧислоСимволов>)

Прав – оставляет нужно количество символов справа. Имеет следующий синтаксис: Лев (<Строка>,<ЧислоСимволов>)

Сред – оставляет нужное количество символов в строке. Имеет следующий синтаксис: Лев (<Строка>,<НачальныйНомер>, <ЧислоСимволов>).

```
СтрокаФИО = "Иванов Иван Иванович"; СтрокаО = Прав(СтрокаФИО,8);
СтрокаФ = Лев(СтрокаФИО,6); СтрокаИ = Сред(СтрокаФИО,8,4);
Сообщить(СтрокаФ); Сообщить(СтрокаИ); Сообщить(СтрокаО);
```

Строки в ЯЗЫКЕ 1С

НАЙТИ В СТРОКЕ 1С

Иногда нужно найти в строке или нужный символ, или нужную группу символов. Для этих целей применятся функция СтрНайти(). Эта функция имеет следующий синтаксис.

СтрНайти
(<Строка>,<ПодстрокаПоиска>,<НаправлениеПоиска>,<НачальнаяПозиция>,<НомерВхождения>)

Данная функция возвращает позицию первого знака подстроки, которая была найдена. Если 0, то ни чего не найдено.

Параметры:

Строка – строка, по которой осуществляется поиск;
ПодстрокаПоиска – подстрока (или символ), которая ищется в строке поиска;
НаправлениеПоиска – системное перечисление, которое задает в какую сторону осуществляется поиск. Имеет два значения: НаправлениеПоиска.Сначала, НаправлениеПоиска. Сконца. Необязательный параметр.
НачальнаяПозиция – номер символа, с которого начинается поиск. Должен быть в диапазоне от 1 до количества символов, иначе будет ошибка. Необязательный параметр. Если он не задан и установлен параметр НаправлениеПоиска, то в случае поиска Сначала по умолчанию равен 1, а если поиск СКонца, то по умолчанию равен количеству символов в строке.
НомерВхождения – искомая подстрока (или символ) может несколько раз входить в исходную строку, этот параметр указывает, какое вхождение нас интересует. По умолчанию равен 1.

Строки в ЯЗЫКЕ 1С

```
СтрокаФИО = "Иванов Сидоров Иванов"; Сообщить(СтрокаФИО); //ищем с настройками по умолчанию НомерИванова = СтрНайти(СтрокаФИО,"Иванов"); Сообщить("Номер подстроки ""Иванов"" в строке " + НомерИванова); //ищем с конца НомерИванова = СтрНайти(СтрокаФИО,"Иванов",НаправлениеПоиска.СКонца); Сообщить("Номер подстроки ""Иванов"" в строке " + НомерИванова); //ищем с конца но начиная с 10 символа НомерИванова = СтрНайти(СтрокаФИО,"Иванов", НаправлениеПоиска.СКонца,10); Сообщить("Номер подстроки ""Иванов"" в строке " + НомерИванова); //ищем с начала, но второе вхождение НомерИванова = СтрНайти (СтрокаФИО,"Иванов",НаправлениеПоиска.СНачала,,2); Сообщить("Номер подстроки ""Иванов"" в строке " + НомерИванова);
```

Сообщения:

- Иванов Сидоров Иванов
- Номер подстроки "Иванов" в строке 1
- Номер подстроки "Иванов" в строке 16
- Номер подстроки "Иванов" в строке 1
- Номер подстроки "Иванов" в строке 16

ЗАМЕНИТЬ В СТРОКЕ 1С

В платформе 1С 8.3. имеется метод, при помощи которого можно менять в строке определенные символы на другие символы.
Этот метод СтрЗаменить, и он имеете следующий синтаксис: СтрЗаменить (<Строка>,<СтрокаПоиска>,<СтрокаЗамены>)
Данные метод возвращается строку, в которой будет выполнена замена или нет, в зависимости от того найдена строка поиска или нет.

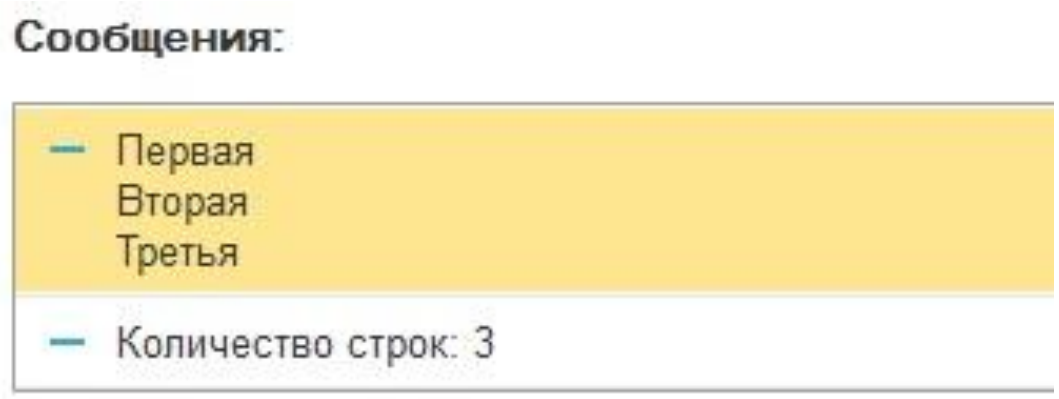
```
ФИОНеПравильное = "Иванов_Андрей_Игоревич"; Сообщить
(ФИОНеПравильное); ФИОВерное = СтрЗаменить(ФИОНеПравильное,"_","
"); Сообщить(ФИОВерное); ФИОБезОтчетства = СтрЗаменить
(ФИОВерное," Игоревич",""); Сообщить(ФИОБезОтчетства);
```

ФУНКЦИИ ДЛЯ МНОГОСТРОЧНЫХ СТРОК В 1С

Из предыдущей статьи вы знаете, что в 1С можно задать многострочную строку, делается это при помощи символа «|». Сейчас мы разберем несколько функций, которые могут пригодиться при работе с многострочной строкой.

СтрЧислоСтрок(<Строка>) – позволяет узнать, сколько в строке строк.

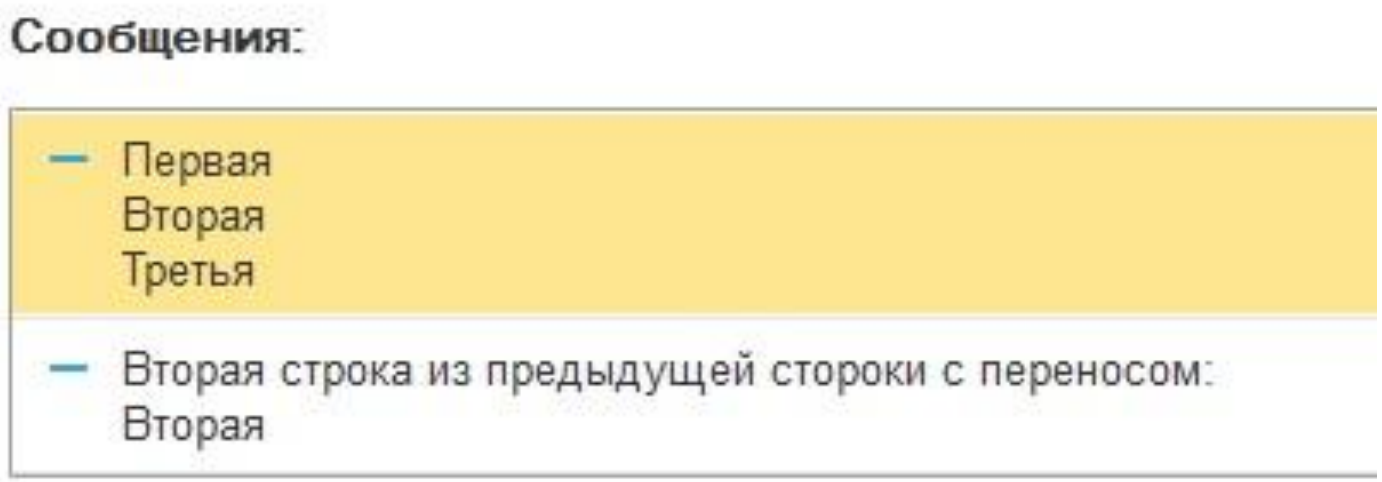
```
СтрокаСПереносом = "Первая |Вторая |Третья"; КоличествоСтрок =
СтрЧислоСтрок(СтрокаСПереносом); Сообщить(СтрокаСПереносом);
Сообщить("Количество строк: " + КоличествоСтрок);
```



Строки в ЯЗЫКЕ 1С

СтрПолучитьСтроку – позволяет получить строку из многострочной строки по номеру, имеете следующий синтаксис: СтрПолучитьСтроку(<Строка>, <НомерПолучаемойСтроки>). <НомерПолучаемойСтроки> — начинается с единицы.

```
СтрокаСПереносом = "Первая |Вторая |Третья"; ВтораяСтрока =
СтрПолучитьСтроку(СтрокаСПереносом,2); Сообщить
(СтрокаСПереносом); Сообщить("Вторая строка из
предыдущей стороки с переносом: |" + ВтораяСтрока);
```



Соответствие и структура в ЯЗЫКЕ 1С

Структура

Представляет собой поименованную коллекцию, состоящую из пар ключ — значение. Ключ может быть только строковым, значение — произвольного типа. К элементу структуры можно обращаться по значению его ключа, т. е. по имени. Обычно используется для хранения небольшого количества значений, каждое из которых имеет некоторое уникальное имя.

Соответствие

Также как и структура, представляет собой коллекцию пар ключ — значение. Однако, в отличие от структуры, ключ может быть практически любого типа.

Что понимается под словом «Ключ»? Ключ – уникален в пределах коллекции и не может иметь двух одинаковых значений.

При попытке поместить по одному и тому же ключу отличные значения система заменит старое значение новым, а не добавит еще один ключ.

Соответствие и структура в ЯЗЫКЕ 1С

Сравнение

Структура	Соответствие
Элементы коллекции:КлючИЗначение	
Свойства: ()	Возможно обращение к значению элемента посредством оператора [...]. В качестве аргумента передается значение ключа элемента.
Для объекта доступен обход коллекции посредством оператора Для каждого … Из … Цикл. При обходе выбираются элементы коллекции.	
Методы:Вставить (Insert) Количество (Count) Очистить (Clear) Удалить (Delete)	
Свойство (Property)	Получить (Get)
Конструкторы:	
По ключам и значениям	По умолчанию
Описание:	

Соответствие и структура в ЯЗЫКЕ 1С

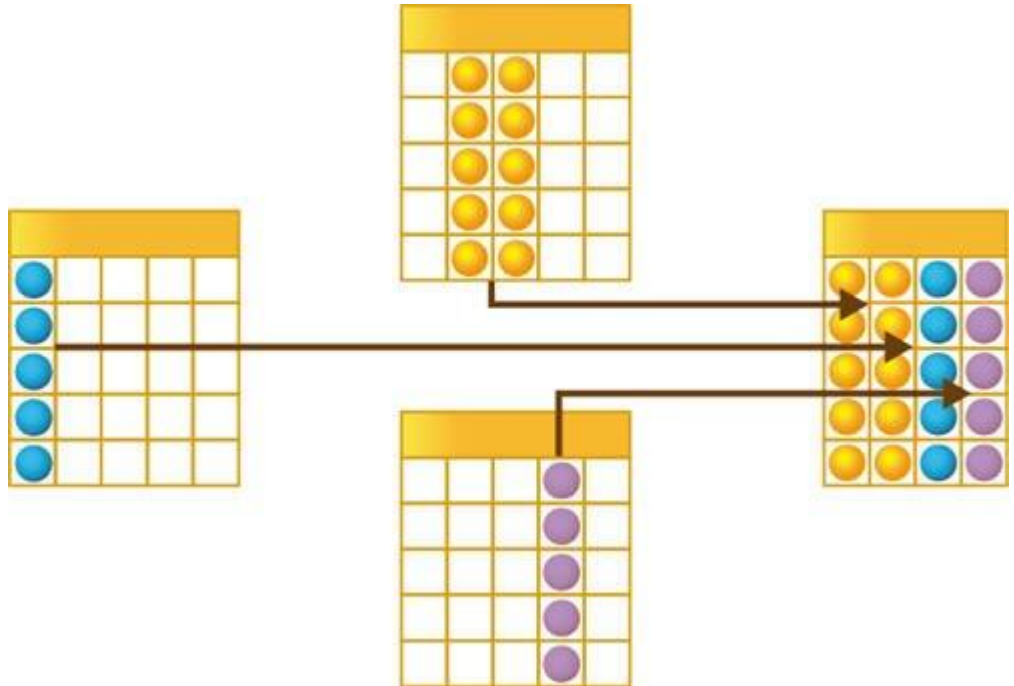
Представляет собой коллекцию пар КлючИЗначение. При этом ключ может быть только строковым и должен удовлетворять требованиям, предъявляемым к именованию переменных встроенного языка. К значениям структуры можно обращаться как к свойствам объекта. При этом ключ используется как имя свойства. Структура используется обычно для хранения небольшого количества значений, каждое из которым имеет некоторое имя.	Представляет доступ к соответствию.
Доступность: Тонкий клиент, веб-клиент, сервер, толстый клиент, внешнее соединение. Возможен обмен с сервером. Сериализуется. Данный объект может быть сериализован в/из XDTO. Тип XDTO, соответствующий данному объекту, определяется в пространстве имен.	
Имя типа XDTO: Structure	Имя типа XDTO: Map Может использоваться в реквизитах управляемой формы.
Пример:	
`Запись = Новый Структура;`	`Запись = Новый Соответствие;`
`Запись.Вставить("Ключ","Значение");`	

Запросы в ЯЗЫКЕ 1С

Механизм запросов — это один из способов доступа к данным, которые поддерживает платформа.
Используя этот механизм, разработчик может
-читать и обрабатывать данные, хранящиеся в информационной базе;
-изменение данных с помощью запросов невозможно.
Это объясняется тем, что запросы специально предназначены для быстрого получения и обработки некоторой выборки из больших массивов данных, которые могут храниться в базе данных.

Табличный способ доступа к данным

Запросы реализуют табличный способ доступа к данным, которые хранятся в базе данных. Это означает, что все данные представляются в виде совокупности связанных между собой таблиц, к которым можно обращаться как по-отдельности, так и к нескольким таблицам во взаимосвязи:
Такой способ работы с данными позволяет получать сложные выборки данных, сгруппированные и отсортированные определенным образом. Для этих выборок могут быть рассчитаны общие и промежуточные итоги, наложены ограничения на количество или состав записей и пр.



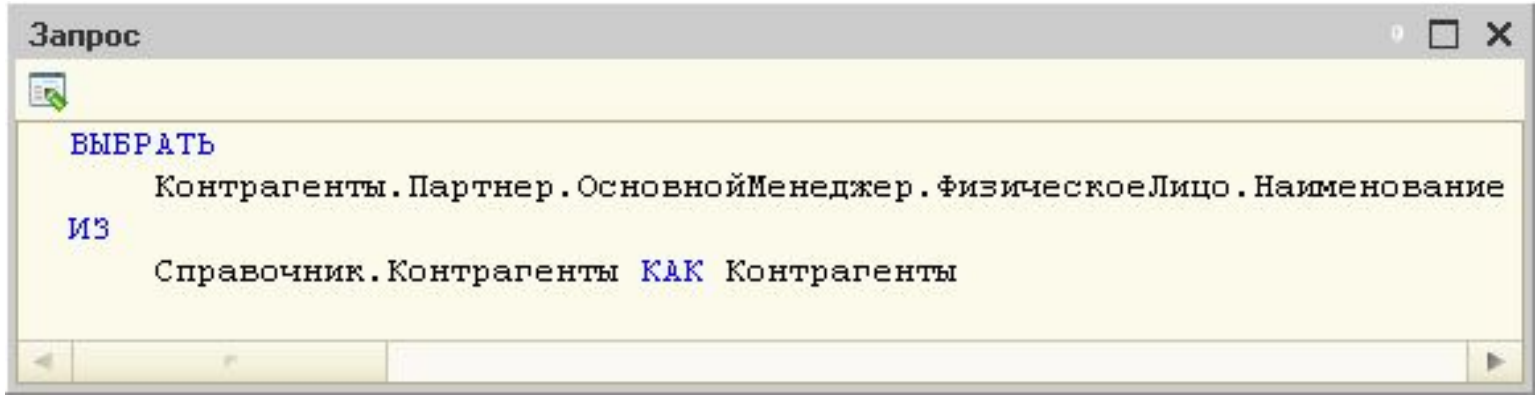
Запросы в ЯЗЫКЕ 1С

Язык запросов

Для того чтобы разработчик имел возможность использовать запросы для реализации собственных алгоритмов, в платформе реализован язык запросов. Этот язык основан на SQL, но при этом содержит значительное количество расширений, ориентированных на отражение специфики финансово-экономических задач и на максимальное сокращение усилий по разработке прикладных решений. Можно перечислить наиболее существенные возможности, реализуемые языком запросов:

Обращение к полям через точку (».»)

Если поля какой-либо таблицы имеют ссылочный тип (хранят ссылки на объекты другой таблицы),
разработчик может в тексте запроса ссылаться на них через «.», при этом количество уровней вложенности таких ссылок система не ограничивает.



Обращение к вложенным таблицам (табличным частям документов и элементов справочников)

Система поддерживает обращения к вложенным табличным частям и как к отдельным таблицам, и как к целым полям одной таблицы. Например, при обращении к документу Реализация товаров (содержащему табличную часть Товары с составом отгружаемых товаров), мы можем считать табличную часть как отдельную таблицу:

Запрос

```
ВЫБРАТЬ
    РеализацияТоваровУслугТовары.Номенклатура,
    РеализацияТоваровУслугТовары.Количество,
    РеализацияТоваровУслугТовары.Цена,
    РеализацияТоваровУслугТовары.Сумма
ИЗ
    Документ.РеализацияТоваровУслуг.Товары КАК РеализацияТоваровУслугТовары
```

Запросы в ЯЗЫКЕ 1С

Но также мы можем считать заголовочную запись документа, в которой значением поля Товары будут все записи вложенной таблицы, подчиненные этому объекту (документу):

Запрос

```
ВЫБРАТЬ
    РеализацияТоваровУслуг.Товары.(
        Номенклатура,
        Количество,
        Цена,
        Сумма
    )
ИЗ
    Документ.РеализацияТоваровУслуг КАК РеализацияТоваровУслуг
```

Автоматическое упорядочивание

Для выбора наиболее правильного («естественного») порядка вывода информации на экран или в отчет разработчику в большинстве случаев достаточно задать режим автоматического упорядочивания.

Запрос

```
ВЫБРАТЬ
    РеализацияТоваровУслугТовары.Номенклатура,
    РеализацияТоваровУслугТовары.Количество,
    РеализацияТоваровУслугТовары.Цена,
    РеализацияТоваровУслугТовары.Сумма
ИЗ
    Документ.РеализацияТоваровУслуг.Товары КАК РеализацияТоваровУслугТовары
АВТОУПОРЯДОЧИВАНИЕ
```

Запросы в ЯЗЫКЕ 1С

Многомерное и многоуровневое формирование итогов

Итоги и подитоги формируются с учетом группировки и иерархии, обход уровней может выполняться в произвольном порядке с подведением подитогов, обеспечивается корректное построение итогов по временным измерениям.

Запрос

```
ВЫБРАТЬ
    РеализацияТоваровУслугТовары.Номенклатура КАК Номенклатура,
    СУММА(РеализацияТоваровУслугТовары.Количество) КАК Количество,
    РеализацияТоваровУслугТовары.Цена,
    СУММА(РеализацияТоваровУслугТовары.Сумма) КАК Сумма
ИЗ
    Документ.РеализацияТоваровУслуг.Товары КАК РеализацияТоваровУслугТовары

СГРУППИРОВАТЬ ПО
    РеализацияТоваровУслугТовары.Номенклатура,
    РеализацияТоваровУслугТовары.Цена
ИТОГИ
    СУММА(Количество),
    СУММА(Сумма)
ПО
    ОБЩИЕ,
    Номенклатура ИЕРАРХИЯ
АВТОУПОРЯДОЧИВАНИЕ
```

Поддержка виртуальных таблиц

Виртуальные таблицы, предоставляемые системой, позволяют получить практически готовые данные для большинства прикладных решений без необходимости составления сложных запросов. Например, такая виртуальная таблица может предоставить данные по остаткам товаров в разрезе периодов на какой-то момент времени. При этом виртуальные таблицы максимально используют хранимую информацию, например, ранее рассчитанные итоги и т. д.

Запрос

```
ВЫБРАТЬ
    СвободныеОстаткиОстаткиИОбороты.Период,
    СвободныеОстаткиОстаткиИОбороты.Номенклатура,
    СвободныеОстаткиОстаткиИОбороты.Склад,
    СвободныеОстаткиОстаткиИОбороты.СвободноНачальныйОстаток,
    СвободныеОстаткиОстаткиИОбороты.СвободноКонечныйОстаток
ИЗ
    РегистрНакопления.СвободныеОстатки.ОстаткиИОбороты(&НачалоПериода, &КонецПериода, День, , )
        КАК СвободныеОстаткиОстаткиИОбороты
```

Стандартные SQL операции

В языке запросов поддерживаются стандартные для SQL операции, такие, как объединение (Union), соединение (Join) и т. д.

Запрос

```
ВЫБРАТЬ
    СвободныеОстатки.Свободно,
    Номенклатура.Наименование
ИЗ
    РегистрНакопления.СвободныеОстатки КАК СвободныеОстатки
        ВНУТРЕННЕЕ СОЕДИНЕНИЕ Справочник.Номенклатура КАК Номенклатура
        ПО СвободныеОстатки.Номенклатура = Номенклатура.Ссылка
```

Запрос

```
ВЫБРАТЬ
    СвободныеОстатки.Свободно,
    Номенклатура.Наименование
ИЗ
    Справочник.Номенклатура КАК Номенклатура
        ЛЕВОЕ СОЕДИНЕНИЕ РегистрНакопления.СвободныеОстатки КАК СвободныеОстатки
        ПО СвободныеОстатки.Номенклатура = Номенклатура.Ссылка
```

Запросы в ЯЗЫКЕ 1С

Временные таблицы

Язык запросов позволяет использовать в запросах временные таблицы. С их помощью можно повысить производительность запросов, в некоторых случаях снизить количество блокировок и сделать текст запроса более легким для восприятия.

Пакетные запросы

Для более удобной работы с временными таблицами в языке запросов поддерживается работа с пакетными запросами — таким образом, создание временной таблицы и ее использование помещаются в один запрос. Пакетный запрос представляет собой последовательность запросов, разделенных символом «;». Запросы исполняются один за другим. Результатом выполнения пакетного запроса в зависимости от используемого метода будет являться либо результат, возвращаемый последним запросом пакета, либо массив результатов всех запросов пакета в той последовательности, в которой следуют запросы в пакете.

Конструкторы запроса

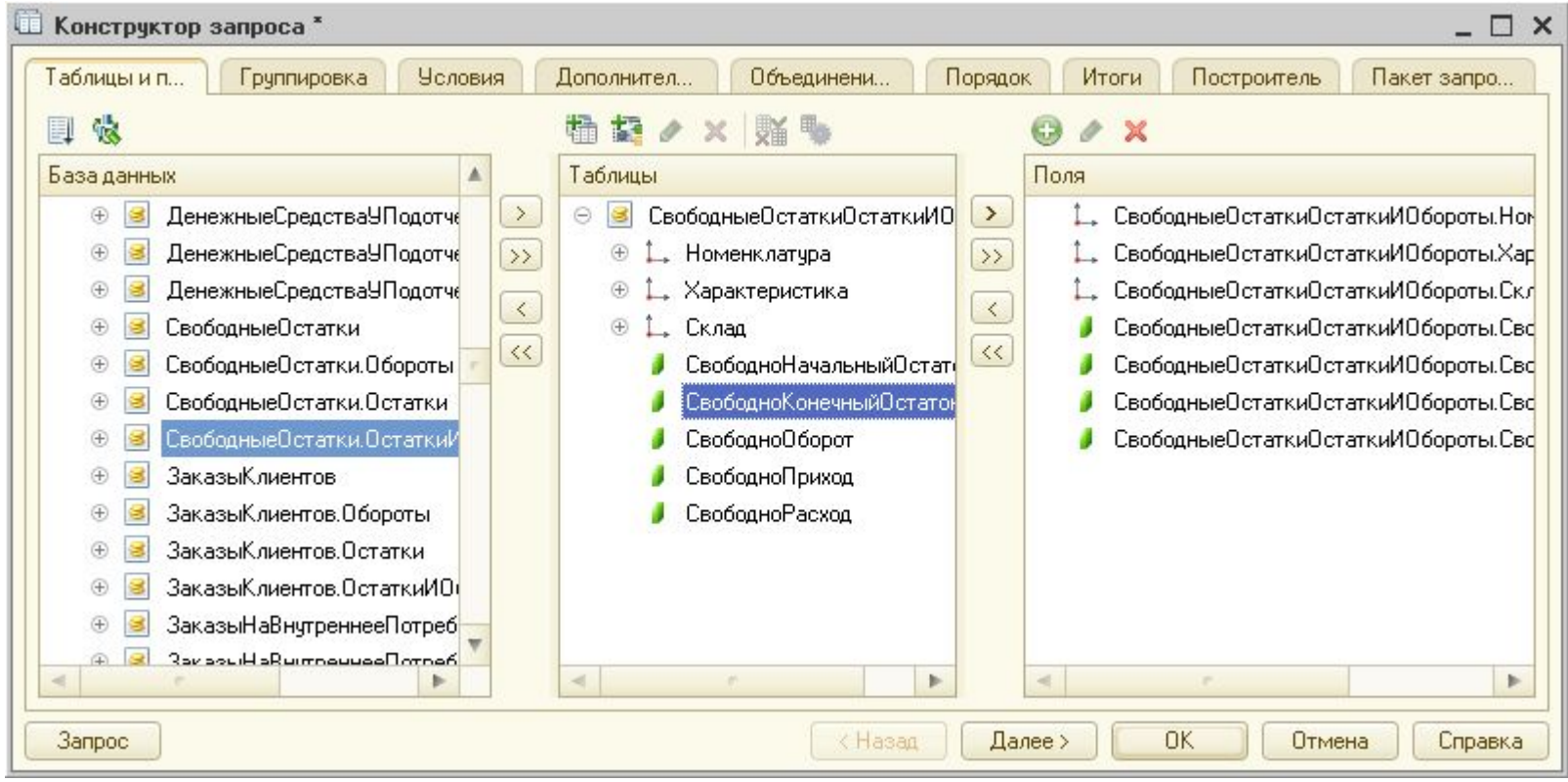
Для облегчения труда разработчика технологическая платформа содержит два специальных конструктора. Они служат для того, чтобы помочь разработчику составить правильный текст запроса, используя только визуальные средства. Выбирая мышью нужные поля таблиц, разработчик может составить работоспособный запрос, даже не зная синтаксиса языка запросов.

Конструктор запроса позволяет составить только текст запроса. Конструктор запроса с обработкой результата помимо текста запроса формирует фрагмент программного кода, который исполняет запрос и выводит результаты в табличный документ или диаграмму.

Запросы в ЯЗЫКЕ 1С

Конструктор запроса — это один из инструментов разработки. Он позволяет составить текст запроса на языке запросов исключительно визуальными средствами.

С помощью кнопок **Далее и Назад** можно перемещаться по закладкам конструктора и указывать, какие данные должны присутствовать в результате запроса, как они связаны, сгруппированы, какие итоги следует рассчитать, работать с временными таблицами, редактировать пакет запросов:



Запросы в ЯЗЫКЕ 1С

Результатом работы конструктора будет являться синтаксически правильный текст запроса. Таким образом, разработчик может составить работоспособный запрос, даже не владея синтаксисом языка запросов — необходимые синтаксические конструкции конструктор сгенерирует автоматически. Готовый текст запроса может быть сразу же вставлен в текст модуля или скопирован в буфер обмена.

Кроме этого конструктор запросов позволяет редактировать уже имеющийся в программе текст запроса. Для этого достаточно установить курсор внутри существующего текста запроса и вызвать конструктор. Имеющийся текст запроса будет проанализирован и представлен в конструкторе в виде соответствующих выбранных полей базы данных и набора заданных связей, группировок, условий и т. д.

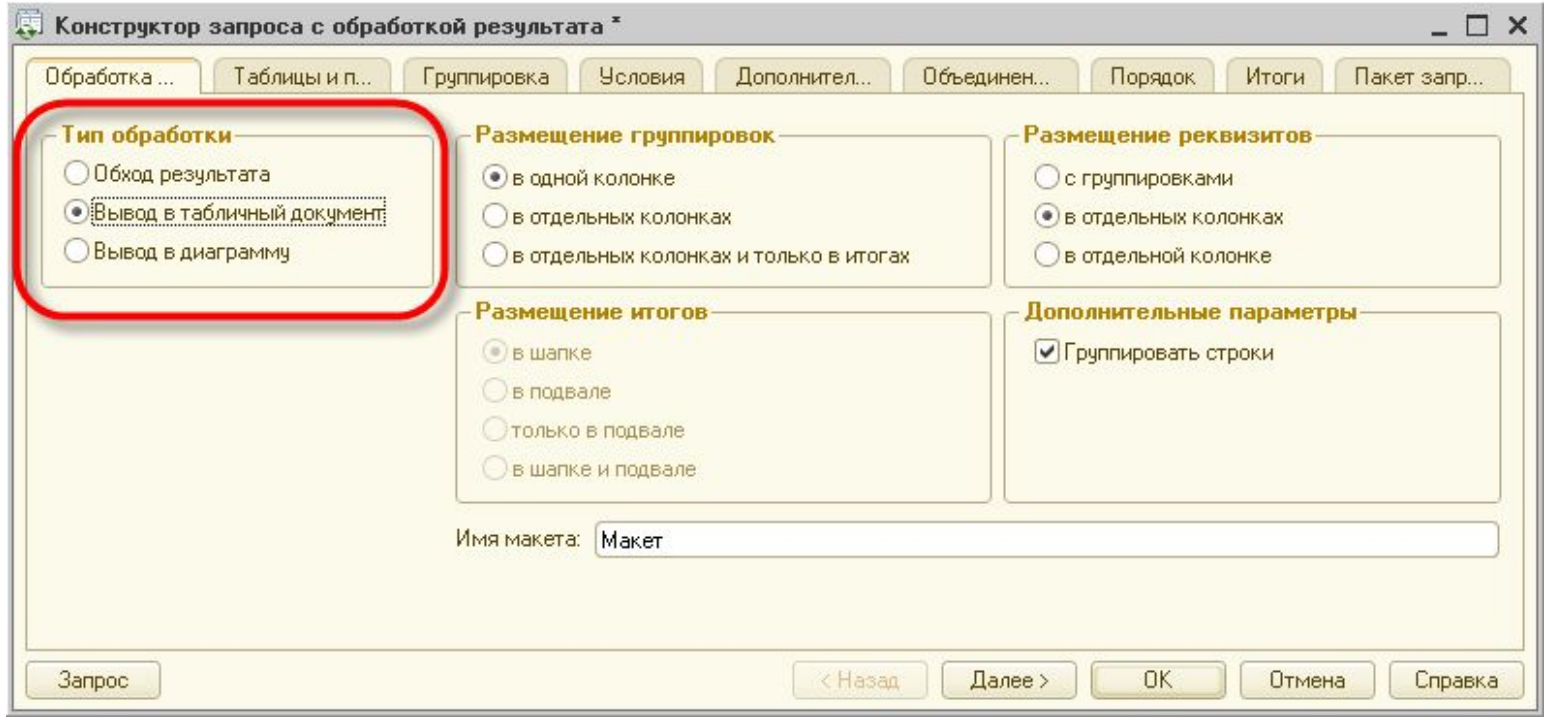
Запрос

```
ВЫБРАТЬ РАЗЛИЧНЫЕ
    ПартнерыКонтактнаяИнформация.Ссылка КАК Партнер,
    ПартнерыКонтактнаяИнформация.Ссылка.Наименование,
    ПартнерыКонтактнаяИнформация.АдресЭП КАК АдресЭП
ИЗ
    Справочник.Партнеры.КонтактнаяИнформация КАК ПартнерыКонтактнаяИнформация
ГДЕ
    ПартнерыКонтактнаяИнформация.АдресЭП ПОДОБНО &СтрокаПоиска

ОБЪЕДИНИТЬ ВСЕ

ВЫБРАТЬ
    КонтактныеЛицаПартнеровКонтактнаяИнформация.Ссылка.Владелец,
    КонтактныеЛицаПартнеровКонтактнаяИнформация.Ссылка.Владелец.Наименование,
    КонтактныеЛицаПартнеровКонтактнаяИнформация.АдресЭП
ИЗ
    Справочник.КонтактныеЛицаПартнеров.КонтактнаяИнформация КАК КонтактныеЛицаПартнеровКонтактна
ГДЕ
    КонтактныеЛицаПартнеровКонтактнаяИнформация.АдресЭП ПОДОБНО &СтрокаПоиска

УПОРЯДОЧИТЬ ПО
    Партнер,
    АдресЭП
ИТОГИ ПО
    Партнер
```

Запросы в ЯЗЫКЕ 1С

Конструктор запроса с обработкой результата — это один из инструментов разработки.
Он позволяет составить текст запроса и сформировать фрагмент программного кода, который исполняет запрос и выводит результаты в табличный документ или диаграмму.
На первом шаге своей работы конструктор предлагает выбрать один из возможных вариантов обработки результата запроса: просто обход результата для его дальнейшей программной обработки или вывод данных в табличный документ или диаграмму.

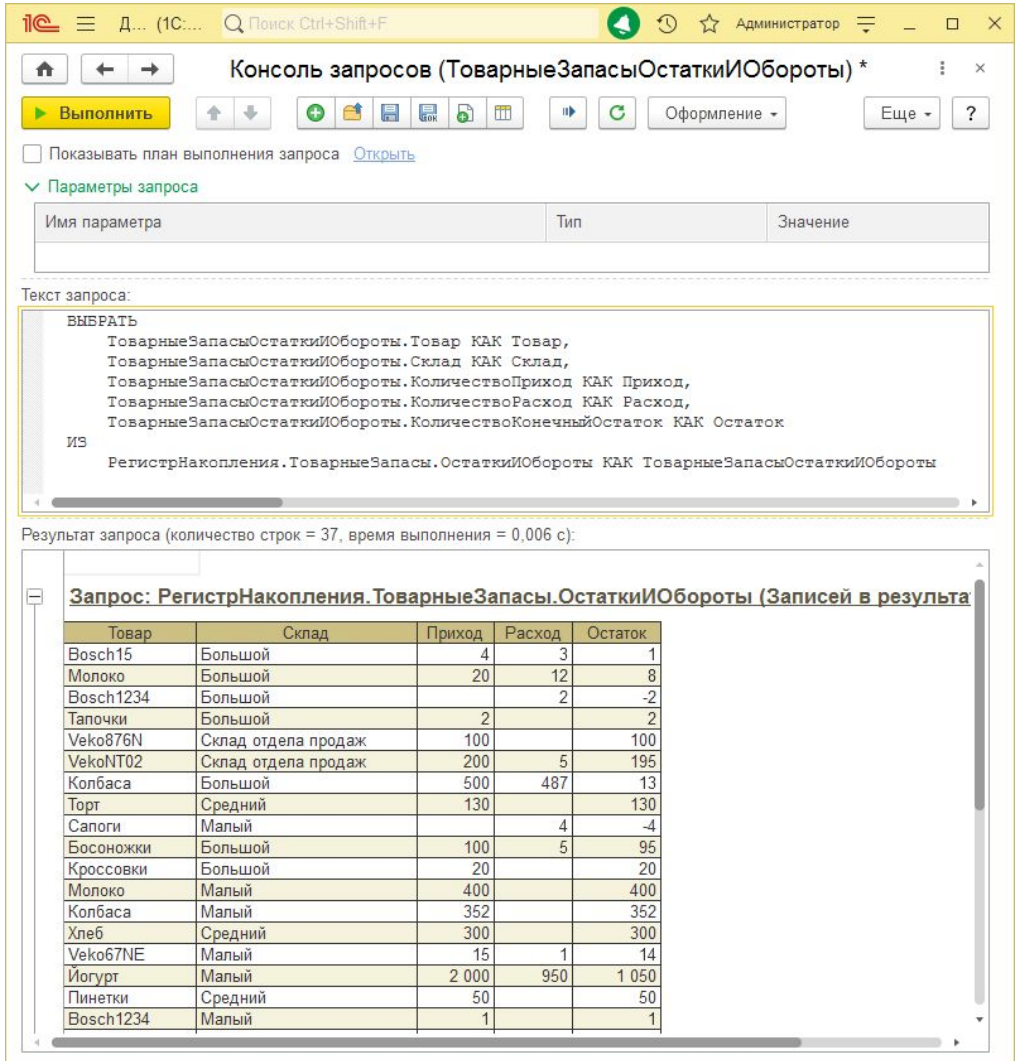
Следующие шаги работы конструктора позволяют создать текст запроса к базе данных. Эти возможности аналогичны тем, которые предоставляет конструктор запроса.



Запросы в ЯЗЫКЕ 1С

Инструмент «Консоль запросов» позволяет разработчикам конфигураций и специалистам по внедрению отлаживать запросы и просматривать результаты их выполнения в режиме «1С: Предприятие 8».

Внешняя обработка Консоль запросов может быть запущен в любом прикладном решении в режиме 1С: Предприятие. С помощью этой обработки разработчик или специалист по внедрению может составить текст запроса, выполнить его и проанализировать полученные результаты:



Запросы в ЯЗЫКЕ 1С

Для того чтобы облегчить составление текста запроса, можно вызвать конструктор запроса, и с его помощью составить запрос любой сложности:

Результат выполнения запроса может быть сохранен в табличный документ для последующего использования. Также консоль запросов позволяет сохранять созданные запросы в файле на диске, если они могут понадобиться в дальнейшем.