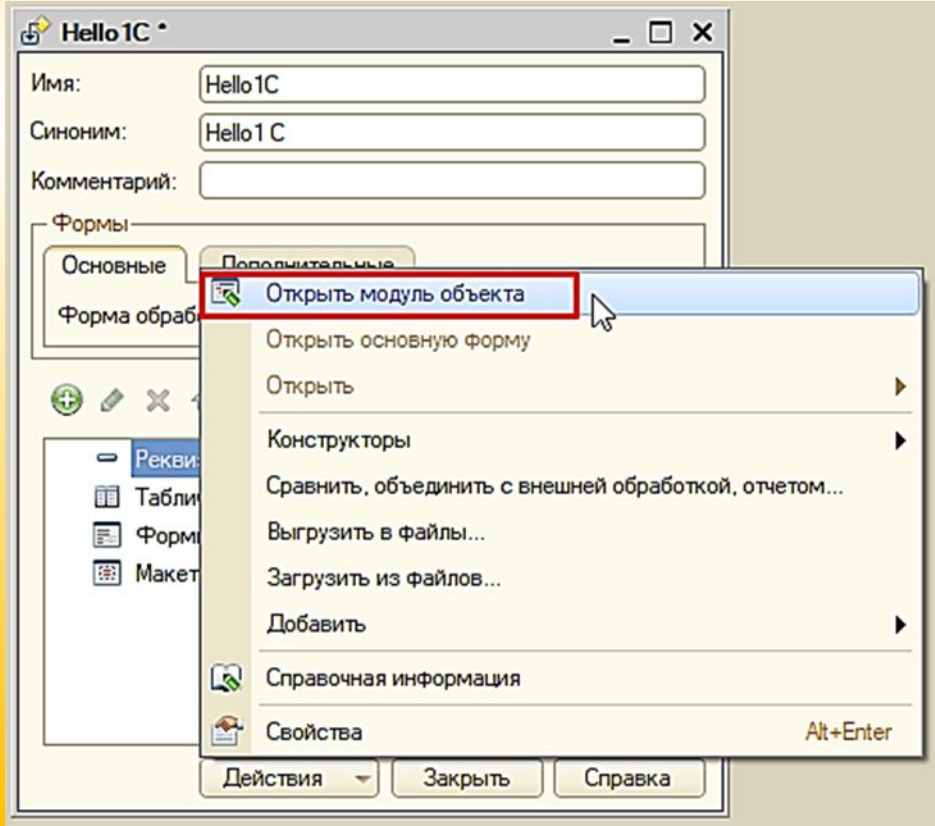
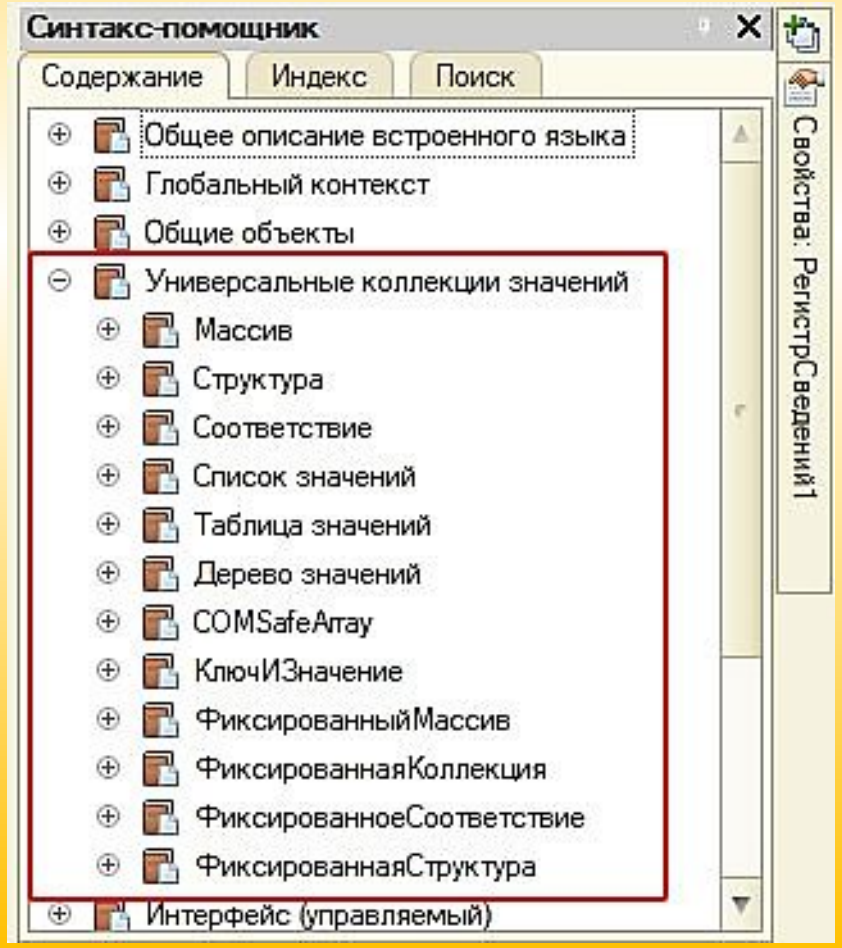
ВСТРОЕННЫЙ ЯЗЫК 1С

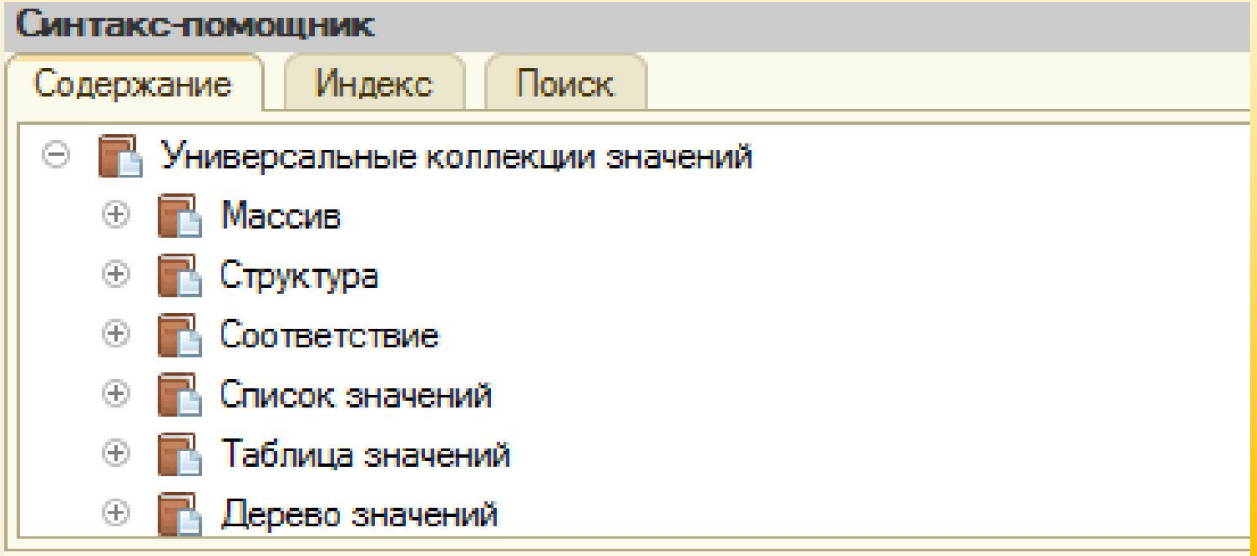


Коллекции значений



Коллекции значений

- Коллекции значений позволяют задать **множество значений, обрабатываемых по единым алгоритмам, как одну переменную.** Разные способы работы с наборами определяют и разнообразие видов коллекций значений:



Массивы

В системе 1С:Предприятие 8

1. Допускается использование как *динамических массивов, размерность которых может изменяться, так и фиксированных*.
2. Нет жестких ограничений на *использование различных типов значений в массиве*
3. Поддерживаются *разные по сложности структуры массивы*. В качестве элементов массива могут выступать, в частности, другие массивы. Это позволяет создавать многомерные массивы.

Массивы

В системе 1С:Предприятие 8

- **Простые (одномерные, линейные) массивы** можно представить в виде таблицы с двумя колонками – индекс и значение элемента:

Индекс	Значение элемента

- **Сложные массивы (многомерные)** могут быть представлены как массивы, значения которых есть другие массивы, т.е. массивы массивов

Индекс	Значение элемента
	<table><tr><th>Индекс</th><th>Значение элемента</th></tr><tr><td></td><td></td></tr></table>
	<table><tr><th>Индекс</th><th>Значение элемента</th></tr><tr><td></td><td></td></tr></table>

Массивы

- *Хранение значения в массиве* осуществляется *по индексу*, представляющему собой целое число (первый элемент имеет индекс 0).

0	3
1	7

- *Работа с элементом массива* осуществляется *по индексу*. Значение индекса задается в квадратных скобках.

Пример: *Для того чтобы обратиться к элементу массива МойМассив, можно использовать обращение по индексу, который указывается в квадратных скобках:*

МойМассив[2]

Так как индекс начинается с нуля, система возвращает элемент массива с индексом 2, но третий (по порядку) элемент массива.

Основные операции массивов

1. Создание массива

СозданныйМассив = Новый Массив();

2. Создание одномерного массива

СозданныйМассив = Новый Массив(10);

3. Создание многомерного массива

ДвумерныйМассив = Новый Массив (2, 8);

4. Создание элементов массива

МоиДанные = Новый Массив(5);

МоиДанные[0] = 3;

МоиДанные[1] = 7;

5. Обращение к элементу массива

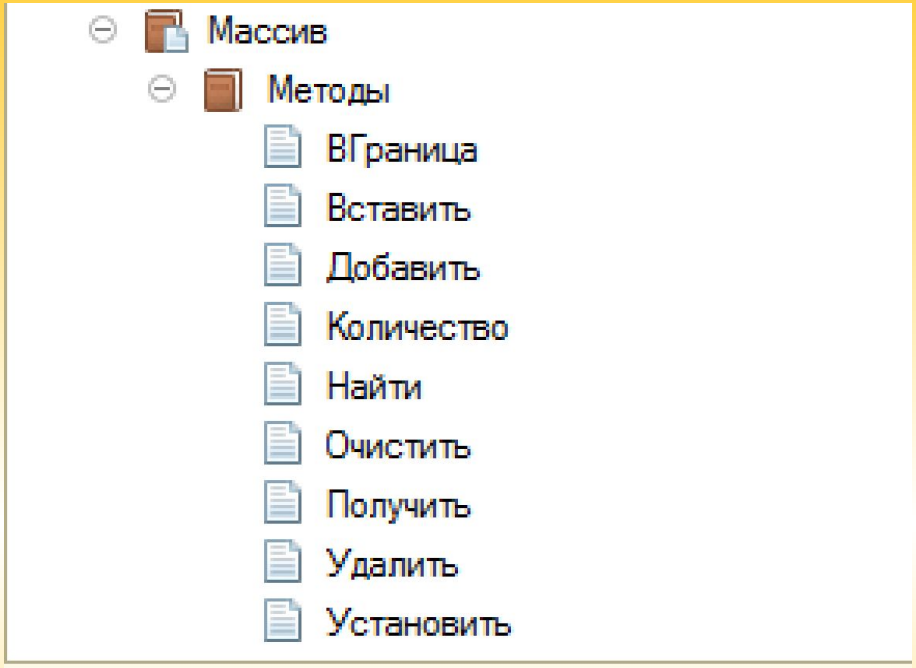
ПервоеЗначение = НашМассив[0];

// ...

ДесятоеЗначение = НашМассив[9];

Массивы

- Для работы с массивами в системе существуют методы:



- Для использования методов с массивами в тексте алгоритма сначала указывается имя переменной, содержащей массив, затем через точку имя метода:

<ИмяПеременной>.<ИмяМетода>

Пример:

```
А = МоиДанные.Количество(); // А=количеству элементов массива
В = МоиДанные.Вграница();     // В=максимальный индекс
```

Пример

Перебор всех числовых элементов массива

```
Для Каждого ЭлементМассива Из МассивЭлементов Цикл
    Если ТипЗнч(ЭлементМассива) = Тип("Число") Тогда
        Сообщить(ЭлементМассива);
    КонецЕсли;
КонецЦикла;
```

Структуры

- При решении некоторых задач есть необходимость *обращения к элементам не по индексам (как в массивах), а по именам*. В этом случае **индекс элементов должен быть строковым**, что реализовано в коллекциях типа **СТРУКТУРА**.

Массив

Индекс	Значение
0	3
1	7

элемент массива

Структура

Ключ	Значение
Первое	3
Второе	7

элемент структуры

Структуры

- **Структура** — поименованная коллекция, состоящая из пар «**ключ — значение**». Обычно используется для хранения ***небольшого количества значений***, каждое из которых имеет некоторое ***уникальное имя***.
- ***Ключ может быть только строковым, значение — произвольного типа***.
- К элементу структуры можно обращаться по ***значению его ключа, т.е. по имени***.

Ключ	Значение
Первое	3
Второе	7

Структуры

- В отличие от массива в структуре **индекс элемента является строковым и называется ключом**. Ключ является *идентификатором элемента в наборе*. Использование имени для каждого элемента набора удобно, если требуется определить структуру хранение элементов с учетом логики дальнейшего их использования.

Например, при хранении адреса:

Адрес= Новый Структура(“Город,Улица,Дом”);

Адрес.Город = “Москва”;

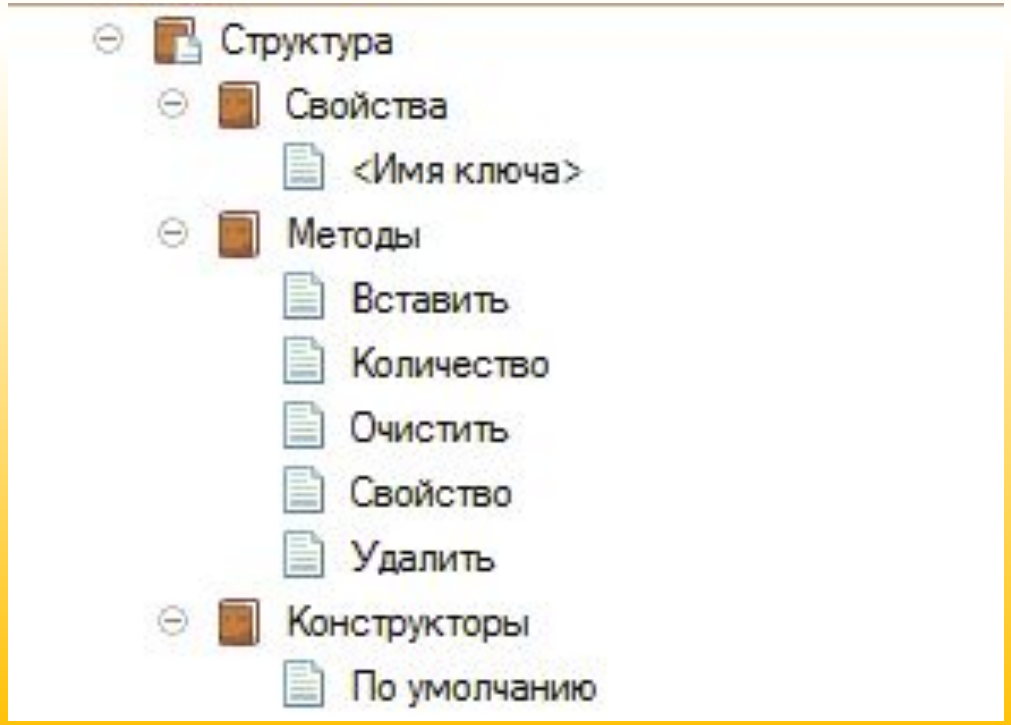
Адрес.Улица = “Лесная”;

Адрес.Дом = 1;

Ключ	Значение
Город	Москва
Улица	Лесная
Дом	1

Структуры

- Возможные **методы и свойства Структуры** описаны в синтакс-помощнике



Основные операции Структуры

1. Создание структуры

СозданнаяСтруктура = Новый Структура;

2. Создание структуры с заранее определенными ключами

Оппонент = Новый Структура (" Обращение, Фамилия, Имя, Отчество, Возраст");

3. Добавление элементов в структуру

Для добавления в структуру элементов используется метод Вставить(), первым параметром указывается ключ, вторым значение нового элемента

Оппонент.Вставить("Обращение", "Господин");

Оппонент.Вставить(" Фамилия ", "Муратов");

Оппонент.Вставить("Возраст", 25);

Основные операции Структуры

4. Чтение элемента структуры с явным указанием ключа

ТекущийВозрат = Оппонент.Возраст;

5. Чтение элемента структуры по ключу, содержащемуся в переменной

КлючСтруктуры = "Возраст";

ТекущийВозраст = Оппонент[КлючСтруктуры];

6. Запись элемента структуры с явным указанием ключа

Оппонет.Возраст = 32;

7. Перебор элементов структуры СтруктураПараметров

```
Для Каждого Элемент из СтруктураПараметров Цикл
    Сообщить(Элемент.Ключ + ": " + Элемент.Значение);
КонецЦикла;
```

Соответствия

- Соответствие также как и структура, представляет собой **коллекцию пар ключ — значение**. Однако, в отличие от структуры, **ключ может быть практически любого типа**.
- Соответствия используются в тех случаях, когда **невозможно использование простого ключа для идентификации элемента** в коллекции. Например:

```
Офисы = Новый Соответствие;
Офисы.Вставить(“Главный офис”, “Лесная 1”);
Офисы.Вставить(“Дополнительный офис”, “Полевая 1”);
```

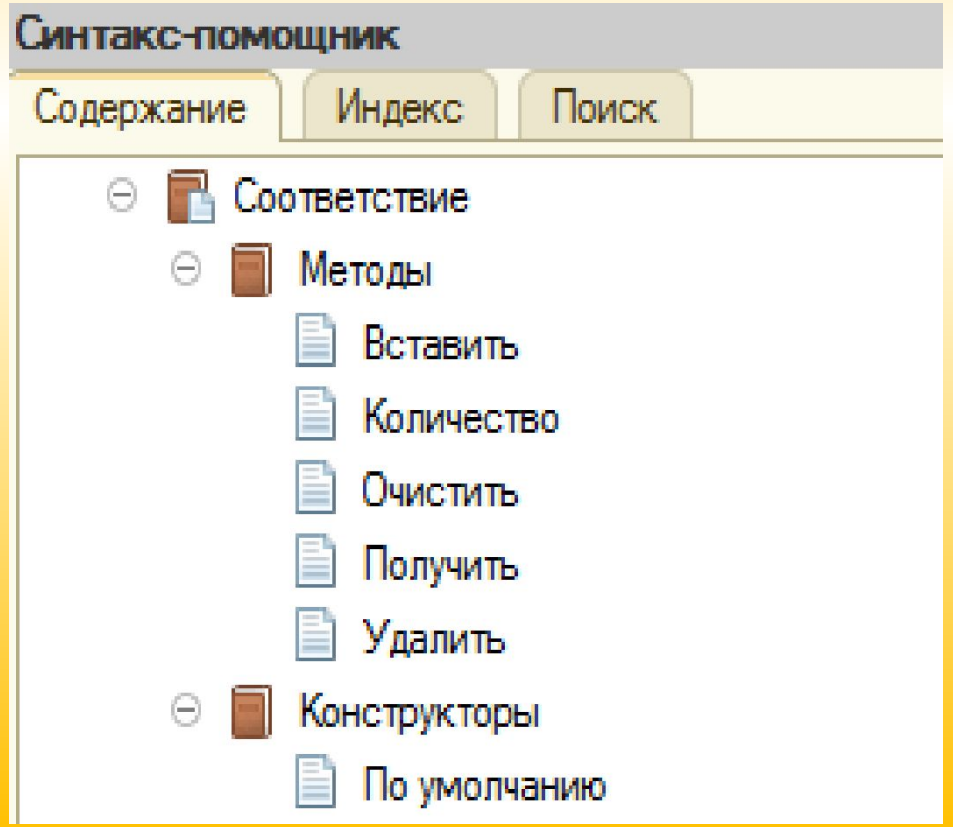
- ***Чтение и редактирование элементов коллекции может производиться с указанием ключа в квадратных скобках*** после имени переменной, в которой хранится соответствие:

```
Офисы[“Главный офис”] = “Лесная 1”;
```

Ключ	Значение
Главный офис	Лесная 1
Дополнительный офис	Полевая 1

Соответствия

- При просмотре списка доступных методов для работы с соответствием в синтакс-помощнике видно, что **отсутствует возможность обращения к элементу коллекции через ключ**. ***Ключ в соответствии не является идентификатором записи, так как может содержать пробелы и может быть не только строковым.***



Соответствия

Например:

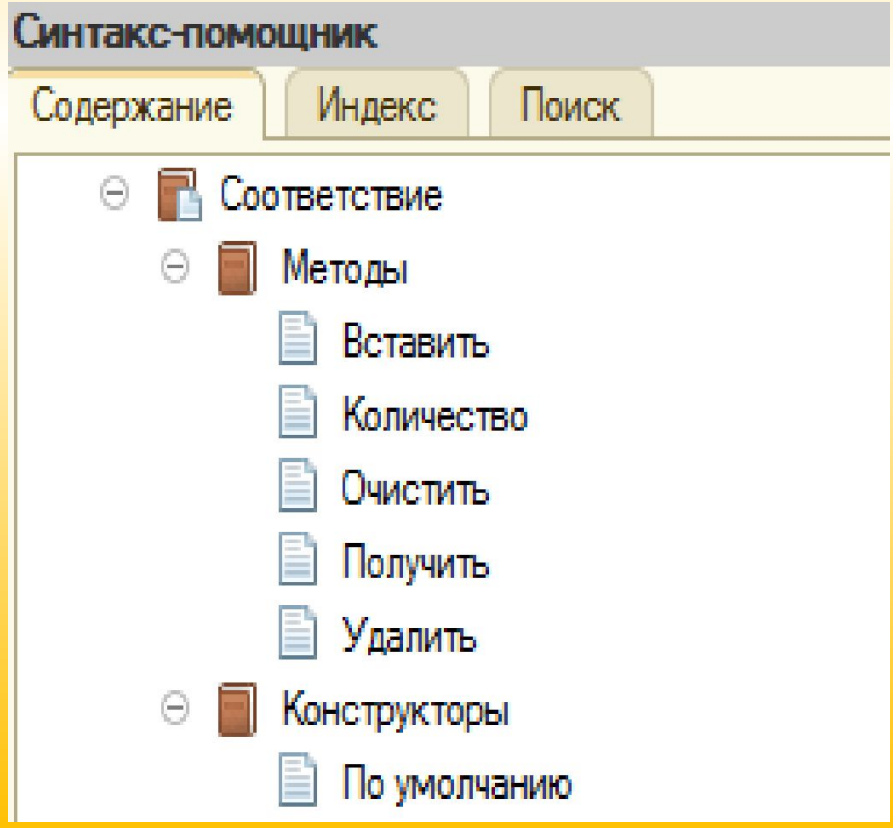
Данные = Новый Соответствие;

Данные.Вставить(0, “Число ноль”);

Данные.Вставить(“0”, “Текст ноль”);

Главным требованием при использовании ключей элементов, является их уникальность в рамках одного соответствия.

Возможные **методы и свойства Соответствия** описаны в синтакс-помощнике



Основные операции Соответствия

1. Создание соответствия

ВозрастСотрудников = Новый Соответствие();

2. Добавление элемента в соответствие с помощью обращения по ключу

ВозрастСотрудников[Сотрудник] = ВозрастСотрудника;

3. Чтение элемента соответствия с помощью обращения по ключу

ВозрастСотрудника = ВозрастСотрудников[Сотрудник];

4. Перебор соответствия Замены

```
Для Каждого Элемент из ВозрастСотрудников Цикл
    Сообщить(Элемент. Сотрудник + " - " + Элемент. ВозрастСотрудника);
КонецЦикла;
```

Списки значений

Список значений предназначен ***для хранения коллекции значений и их пользовательских представлений в интерфейсе*.**

Список значений позволяет строить **динамические наборы значений и манипулировать ими** (добавлять, редактировать, удалять элементы, сортировать).

Используется, как правило, для решения интерфейсных задач.

Особенности списка значений:

- может содержать **значения** любого типа;
- в одном списке **типы хранимых значений** могут быть разными.

Списки значений

Представление значений может задаваться тремя вариантами: **«Представление», «Пометка», «Картинка».**

- **Представление** используется для хранения строкового представления значения и имеет тип «Строка»
- **Пометка** может использоваться, например, для хранения признака использования значения и имеет тип «Булево»
- **Картинка** используется для графического представления значения и имеет тип «Картинка».

Значение	Представление	Пометка	Картинка
77	1С:Предприятие 77	Ложь	
8	1С:Предприятие 8	Истина	

Список значений

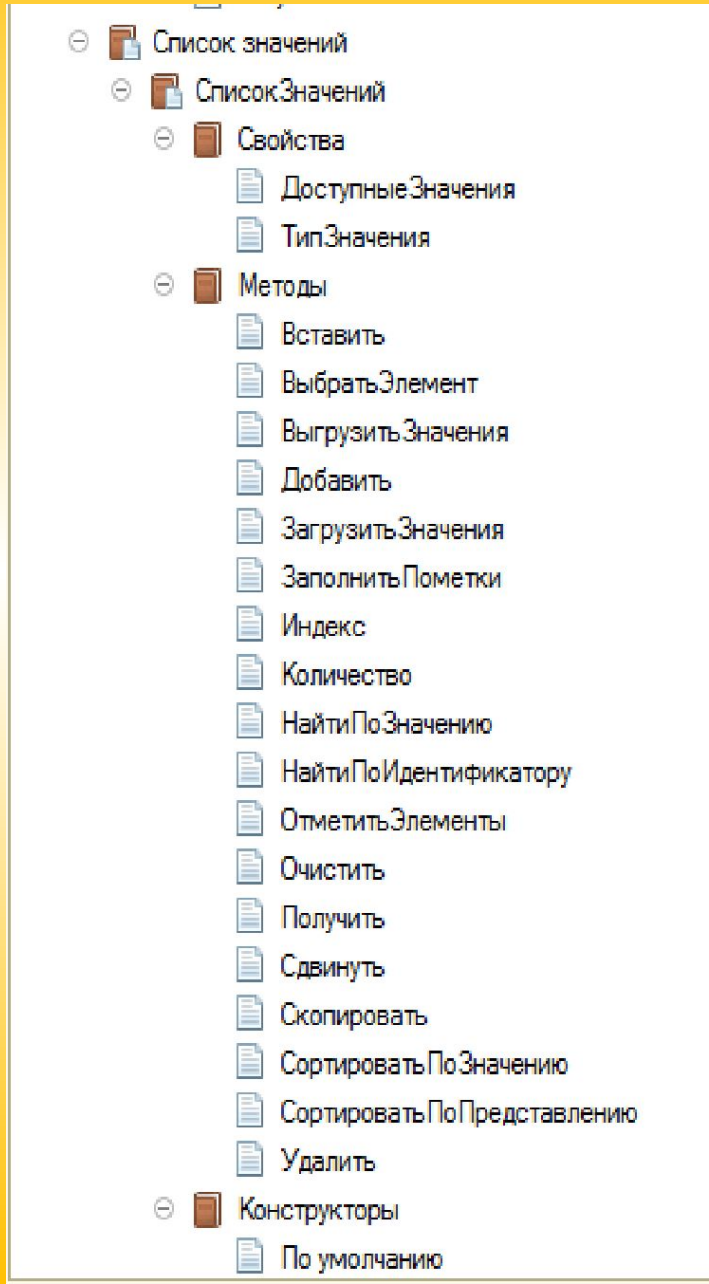
Для работы со списком значений доступны следующие методы и свойства

Пример:

Список1=Новый СписокЗначений;

Список1.Добавить(77,”1С:Предприятие 77”);

Список1.Добавить(82,”1С:Предприятие 82”);



Основные операции Списка значений

1. Создание Списка значений

СозданныйСписок = Новый СписокЗначений;

2. Добавление элемента в Список значений СписокСотрудников нового сотрудника со **значением - ссылкой на сотрудника (ТекущийСотрудник), ФИО и фото**

СписокСотрудников.Добавить(ТекущийСотрудник, ТекущийСотрудник.ФИО, , Фото);

3. Поиск элемента в списке значений по ссылке, хранящейся в переменной ИскомыйСотрудник

```
ЭлементСписка = СписокСотрудников.НайтиПоЗначению (ИскомыйСотрудник);
    Если ЭлементСписка <> Неопределено Тогда
     Сообщить(СписокСотрудников.Индекс(ЭлементСписка));
КонецЕсли;
```

4. Обращение к элементу списка значений по индексу

ЭлементСписка = СписокСотрудников[Инд];

Основные операции Списка значений

5. Получение значений свойств элемента списка значений

```
Сотрудник     = ЭлементСписка.Значение;
Представление = ЭлементСписка.Представление;
Пометка    = ЭлементСписка.Пометка;
Картинка      = ЭлементСписка.Картинка;
```

6. Перебор списка значений в произвольном порядке

```
Для Каждого ЭлементСписка Из СписокСотрудников Цикл
    Сообщить(ЭлементСписка.Значение);
КонецЦикла;
```

7. Перебор списка значений в порядке индексов

```
СтаршийИндекс = СписокСотрудников.Количество() - 1;
 Для Сч = 0 по СтаршийИндекс Цикл
     Сообщить(СписокСотрудников[Сч].Значение);
КонецЦикла;
```

Таблицы значений

Список значений позволяет хранить ***в строке списка только одно значение и варианты его представления***.

Для решения некоторых задач требуется хранить в строке таблицы ***множество значений***. Для реализации таких задач используются таблицы значений

Список значений

Значение	Поля представления		
5	Пять		
2	Два		

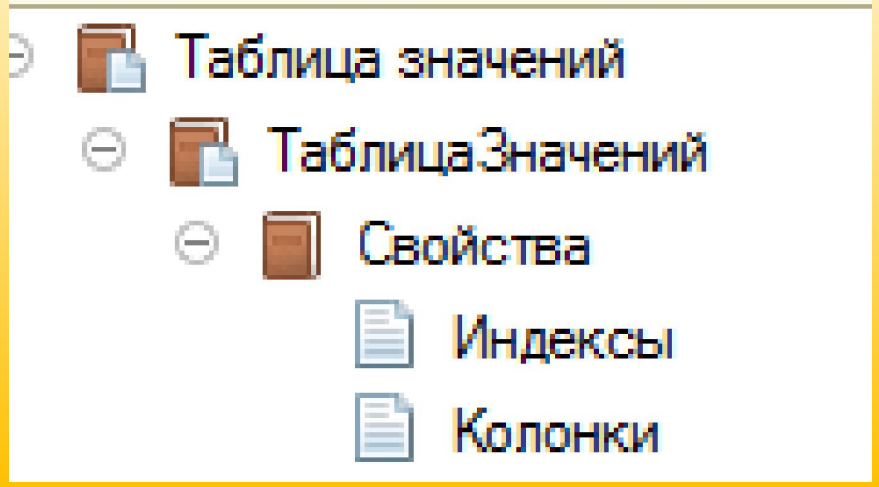
Таблица значений

Значение 1	Значение 2	Значение 3
Ручка	Шариковая	Зеленая
Ручка	Шариковая	Белая

Набор колонок таблицы значений абсолютно произвольный и может определяться и изменяться в любое время.

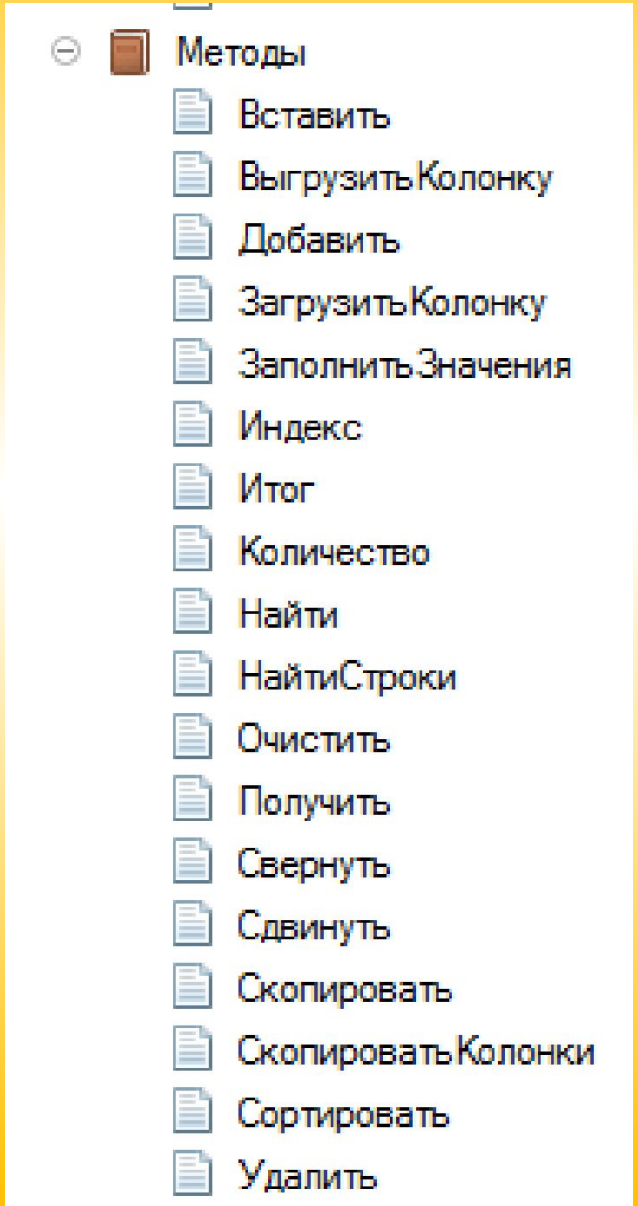
Таблицы значений

- ***При определении значения переменной типа «ТаблицаЗначений» требуется определить структуру колонок таблицы с помощью свойства коллекции «Колонки»,*** что можно увидеть в описании данной коллекции:



Таблицы значений

- Для работы с таблицей значений доступны следующие методы:



Операции с Таблицей значений

1. Создание таблицы значений с определенной структурой хранения данных:

```
Ред1 = Новый ТаблицаЗначений;
Ред1.Колонки.Добавить(“НомерРедакции”);
Ред1.Колонки.Добавить(“НазваниеПродукта”);
```

2. Добавление новой записи:

```
НоваяЗапись = Ред1.Добавить();
НоваяЗапись.НомерРедакции = 77;
НоваяЗапись.НазваниеПродукта = “1С:Предприятие 77”;
```

НомерРедакции	НазваниеПродукта
77	1С:Предприятие 77

Операции с Таблицей значений

3. Перебор таблицы значений в произвольном порядке

```
Для Каждого ТекущаяСтрока Из ТаблицаСотрудников Цикл
    Сообщить(ТекущаяСтрока.Сотрудник);
КонецЦикла;
```

4. Перебор таблицы значений в порядке индексов

```
СтаршийИндекс = ТаблицаСотрудников.Количество() - 1;

Для Сч = 0 по СтаршийИндекс Цикл
    Сообщить(СписокСотрудников[Сч].Сотрудник);
КонецЦикла;
```

Дерево значений

- **Дерево значений** представляет собой динамически формируемый набор значений любого типа. Дерево значений является ***развитием таблицы значений***.
- В отличие от таблицы значений, строки дерева значений могут образовывать ***иерархические структуры***: ***каждая строка дерева может иметь набор подчиненных строк, каждая из подчиненных строк, в свою очередь, также может иметь набор подчиненных строк и так далее.***

Дерево значений

- Пример исходных данных для дерева значений можно представить в виде таблицы:

НомерРедакции	НазваниеПродукта
7	**1С:Предприятие 7**
7.0	1С:Предприятие 7.0
7.5	1С:Предприятие 7.5
7.7	1С:Предприятие 7.7
8	**1С:Предприятие 8**
8.0	1С:Предприятие 8.0
8.1	1С:Предприятие 8.1
8.2	1С:Предприятие 8.2

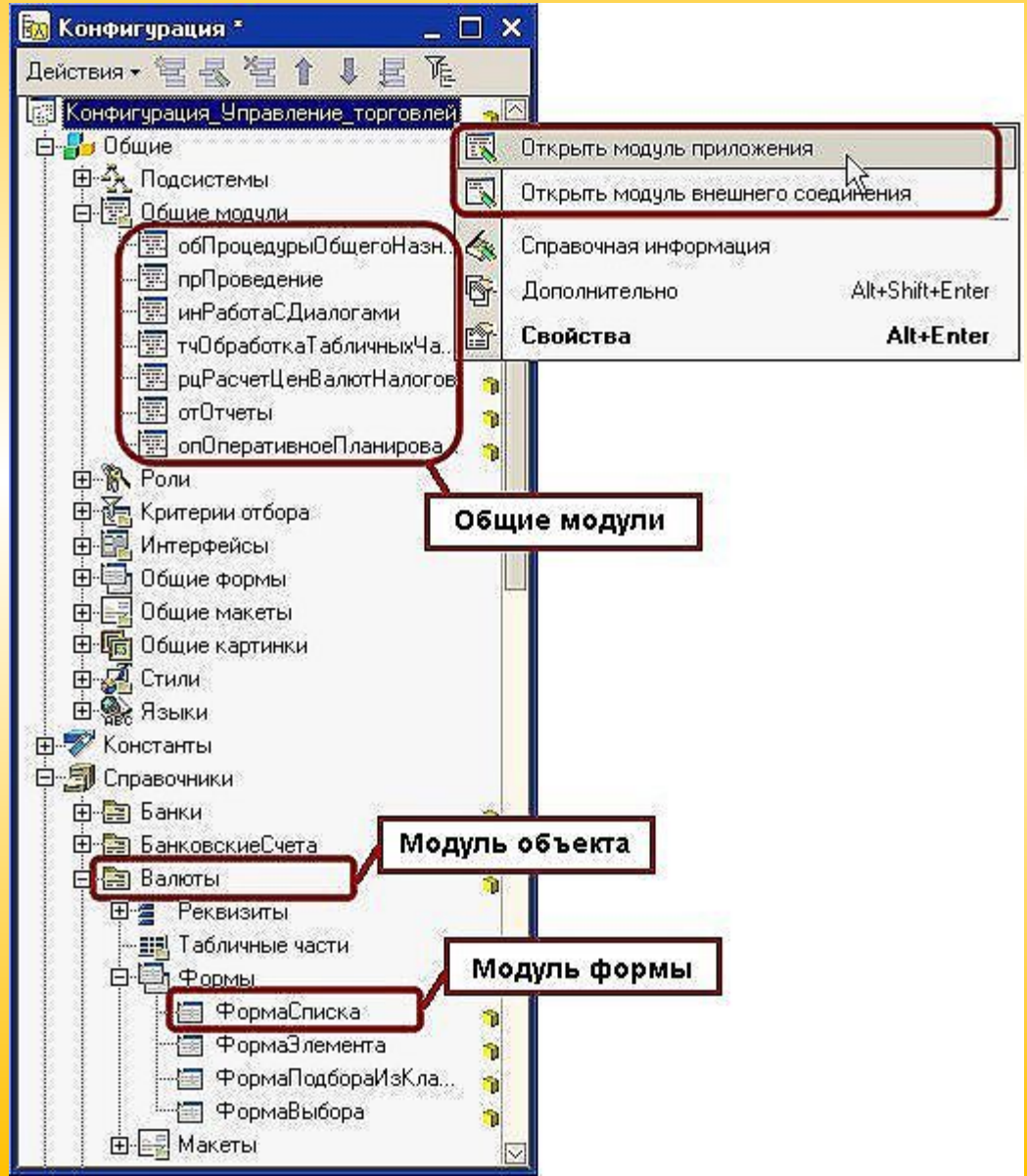
Дерево значений

Пример программного создания и заполнения дерева значений:

```
Ред1С = Новый ДеревоЗначений; //Создание нового дерева значений

Ред1С.Колонки.Добавить(“НомерРедакции”); //Добавление колонок
Ред1С.Колонки.Добавить(“НазваниеПродукта”);

Группа7=Ред1С.Строки.Добавить(); //Добавление строки-родителя
Группа7.НомерРедакции=7;
Группа7.НазваниеПродукта=“1С:Предприятие 7”;

НоваяЗапись=Группа7.Строки.Добавить(); //Добавление строки в группу
НоваяЗапись.НомерРедакции = 7.0;
НоваяЗапись.НазваниеПродукта = “1С:Предприятие 7.0”;
```

Поиск значений, сортировка, получение итогов в Дереве значений могут осуществляться либо по текущему уровню иерархии, либо включая все подчиненные.

Программные модули



Программные модули

Программные модули в конфигурации системы «1С:Предприятие» являются частью конфигурации прикладного решения.

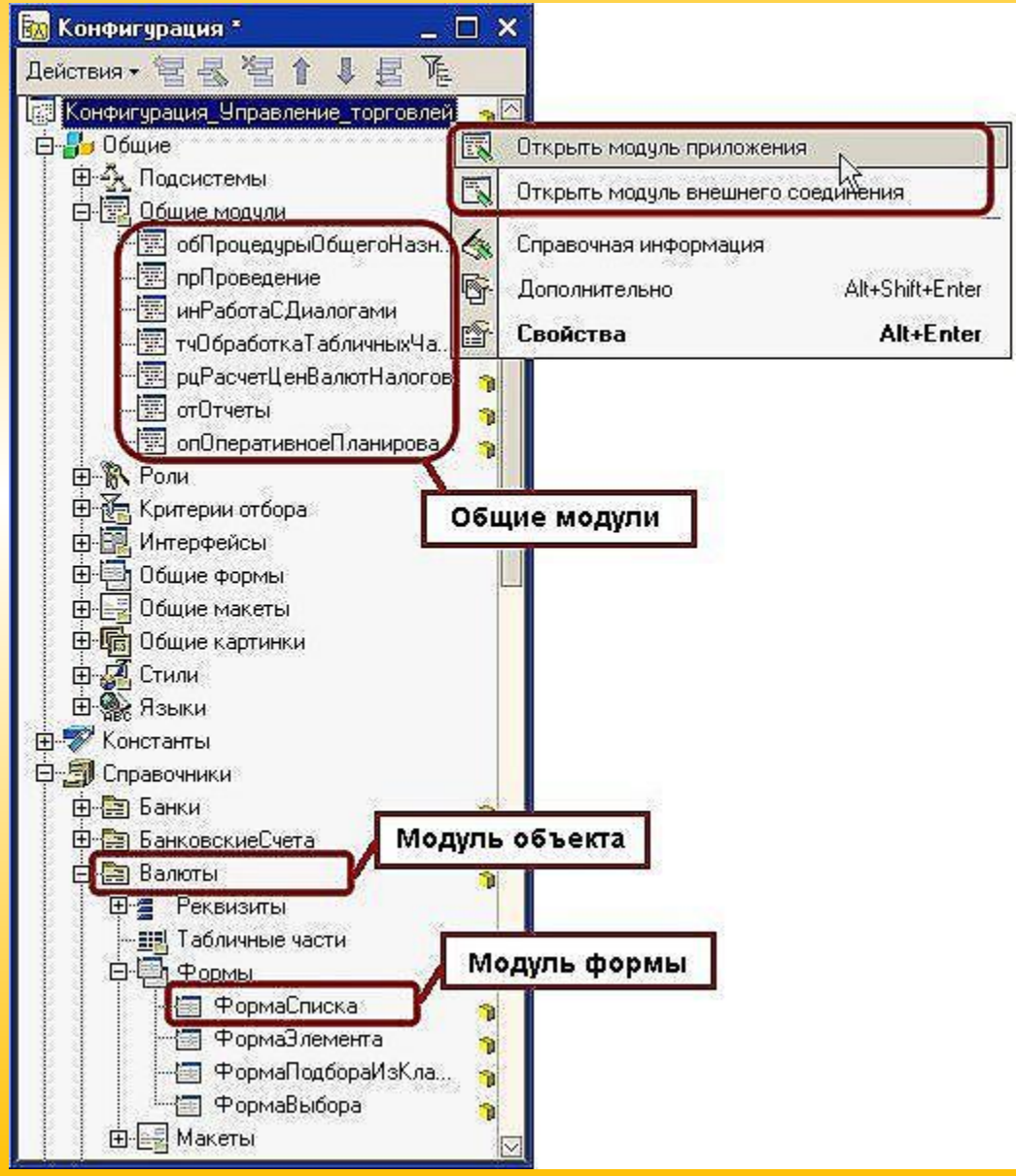
Программный модуль - это ***текст на встроенном языке, в котором размещены тексты процедур и функций с необходимыми алгоритмами, вызываемые системой в определенные моменты работы.***

Поэтому программный модуль **не имеет формальных границ своего описания типа: «Начало модуля» - «Конец модуля».**

Программные модули

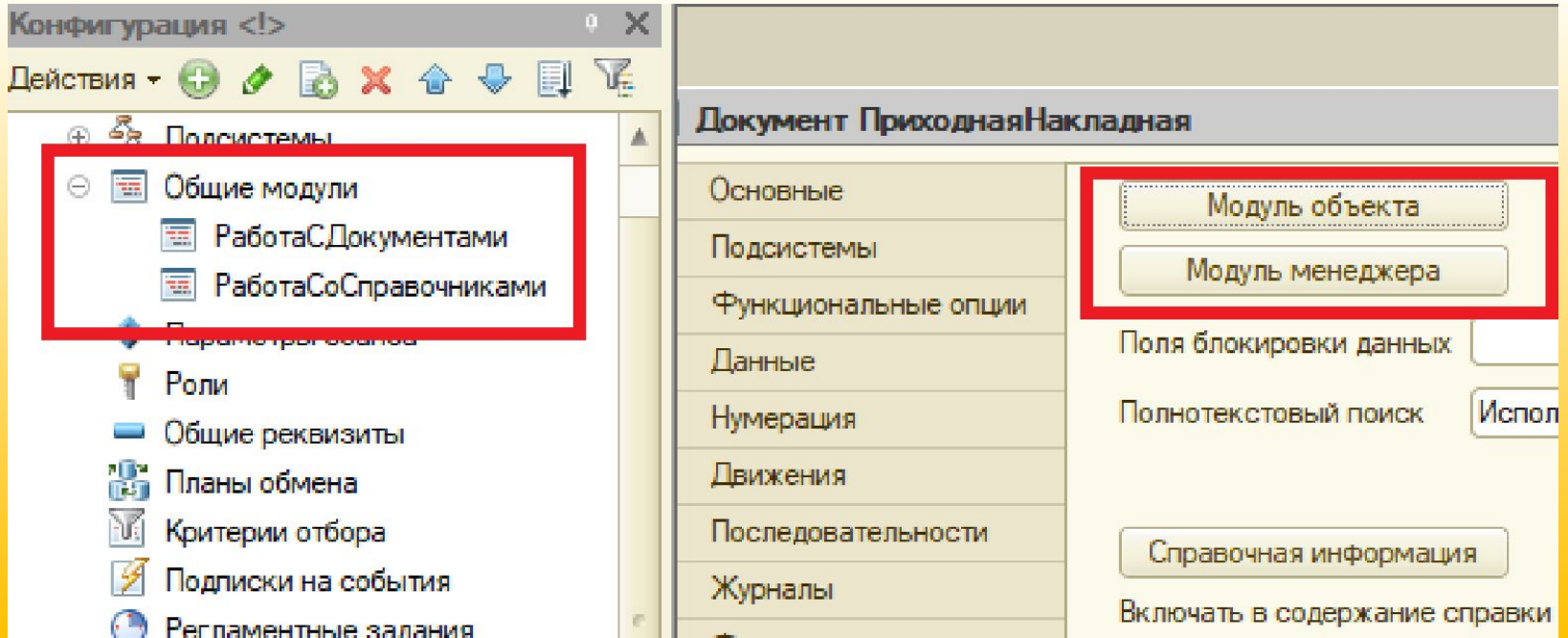
Место размещения конкретного программного модуля предоставляется конфигуратором **в тех точках конфигурации, которые требуют описания специфических алгоритмов функционирования.**

Эти алгоритмы следует ***оформлять в виде процедур или функций, которые будут вызваны самой системой в заранее предусмотренных ситуациях*** (например, при нажатии кнопки в диалоговом окне).



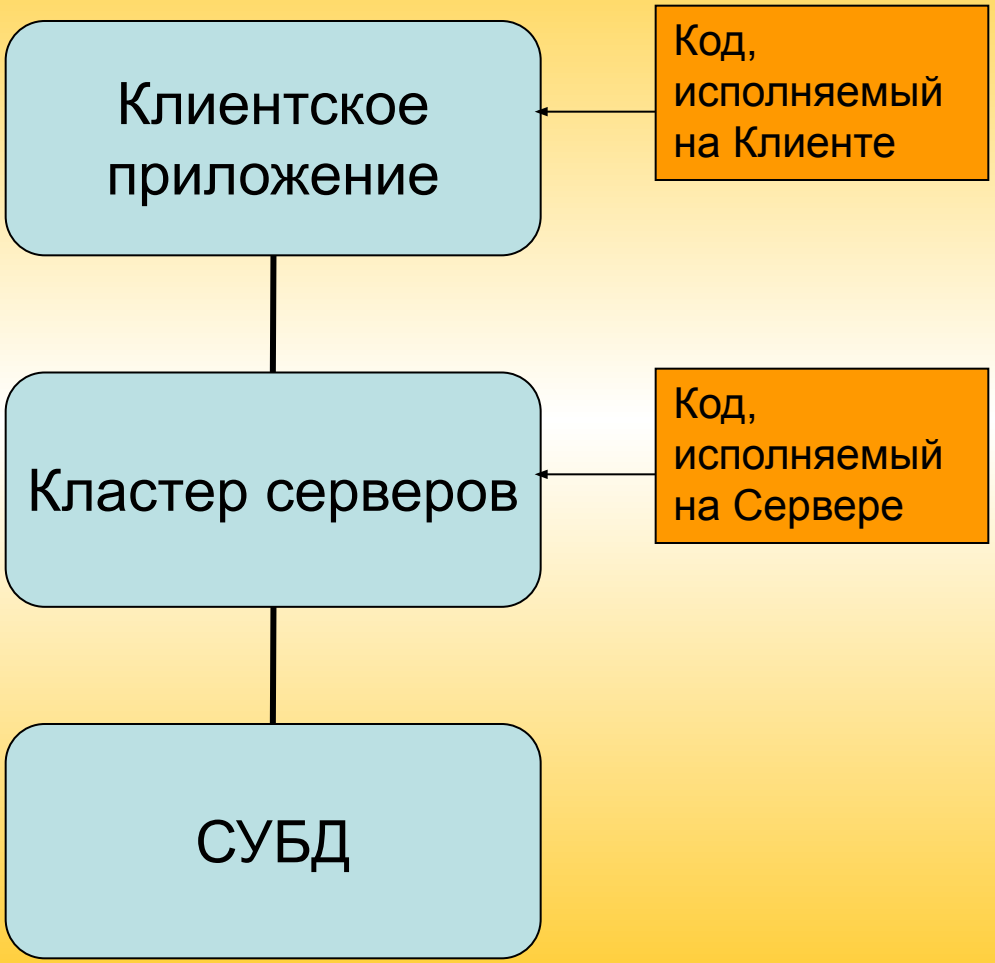
Программные модули

- В Платформе существует достаточно ***большое количество видов модулей, каждый из которых имеет свое предназначение и особенности.***
- Любая строка кода должна находиться в каком-либо модуле. Различают **модули общего предназначения и модули объекта**.



Программные модули

- Некоторые модули могут быть скомпилированы (доступны) **как на Клиенте, так и на Сервере, а некоторые только на Сервере или только на Клиенте**



Формат программного модуля

Структуру программного модуля можно подразделить на следующие разделы:

- ***раздел описания переменных,***
- ***раздел процедур и функций,***
- ***раздел основной программы.***

В конкретном программном модуле любой из разделов может отсутствовать.

Пример структуры программного модуля

```
//****************** ОБЛАСТЬ ОБЪЯВЛЕНИЯ ПЕРЕМЕННЫХ *********************
Перем Фамилия Экспорт; //это глобальная переменная
Перем Имя, Отчество;     //это переменная модуля
Перем ФИО;       //это тоже переменная модуля и к ней можно обращаться
                  //из любой процедуры и функции нашего модуля

//*************** ОБЛАСТЬ ОПИСАНИЯ ПРОЦЕДУР И ФУНКЦИЙ ****************
Процедура Процедура1( )
   Перем Итог;    //Итог это локальная переменная (переменная процедуры)
Итог = Фамилия+" "+Имя+" "+Отчество;

КонецПроцедуры
Функция Функция1( )
   // операторы функции
Возврат(Фамилия + " "+ Имя);
КонецФункции

//****************** ОСНОВНОЙ ТЕКСТ ПРОГРАММЫ ***********************
Фамилия ="Иванов";
Имя = "Иван";
Отчество = "Иванович";
//*****************************************************************************
```

Формат программного модуля

Раздел описания переменных размещается от начала текста модуля до первого оператора Процедура, или оператора Функция, или любого исполняемого оператора. В этом разделе могут находиться только ***операторы объявления переменных Перем.***

В некоторых модулях **для переменных может указываться место компиляции (доступность) на Сервере или на Клиенте**. Например:

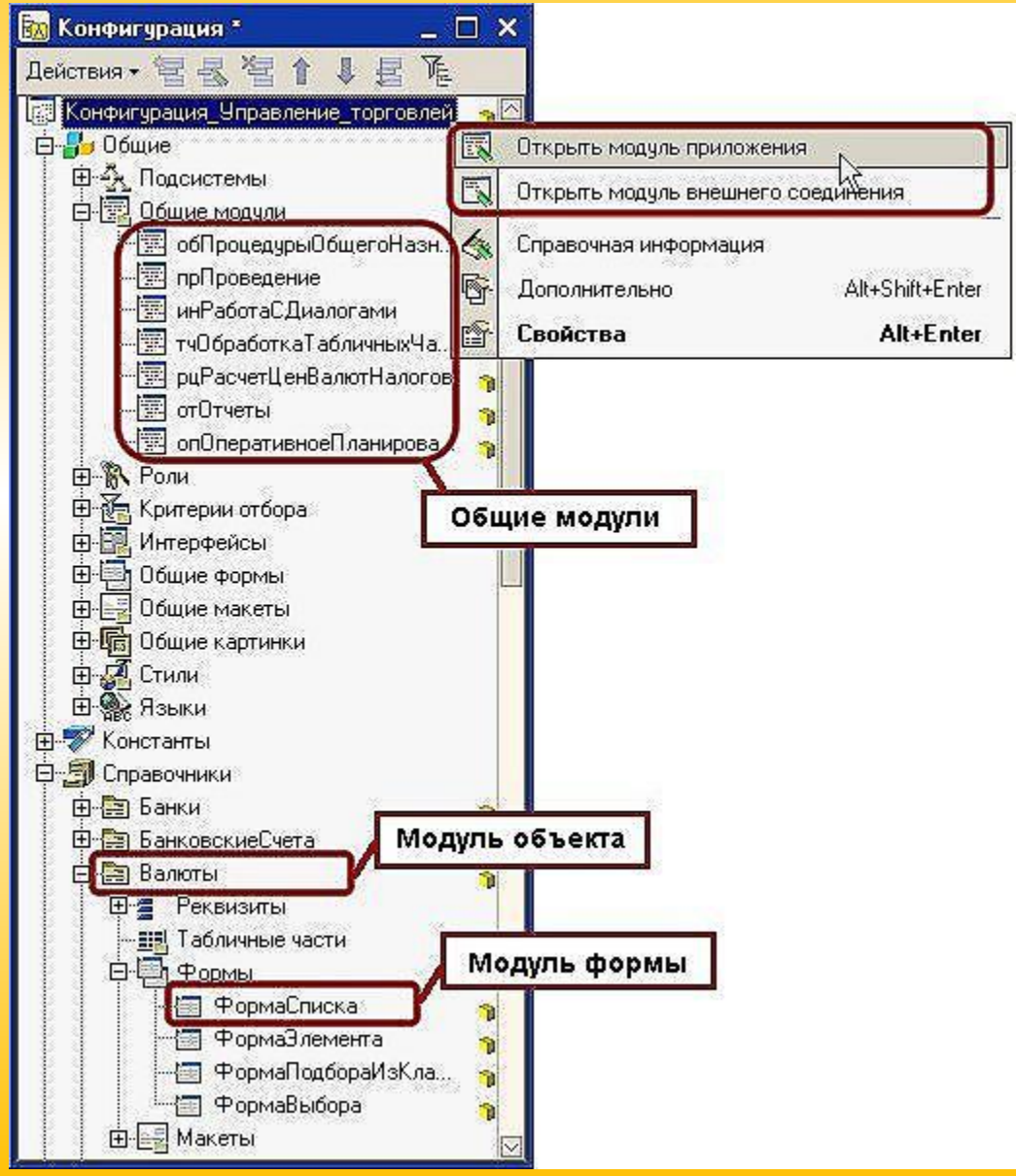
```
&НаСервере
Перем Счетчик;
```

Раздел процедур и функций размещается от первого оператора Процедура или оператора Функция до любого исполняемого оператора вне тела описания процедур или функций.

Раздел основной программы размещается от первого исполняемого оператора вне тела последней процедуры или функции до конца модуля. В этом разделе могут находиться только исполняемые операторы. Раздел основной программы исполняется в момент инициализации модуля.

Виды программных модулей

В системе «1С: Предприятие» существуют несколько видов программных модулей. Они различаются по **месту размещения и функциональности**.



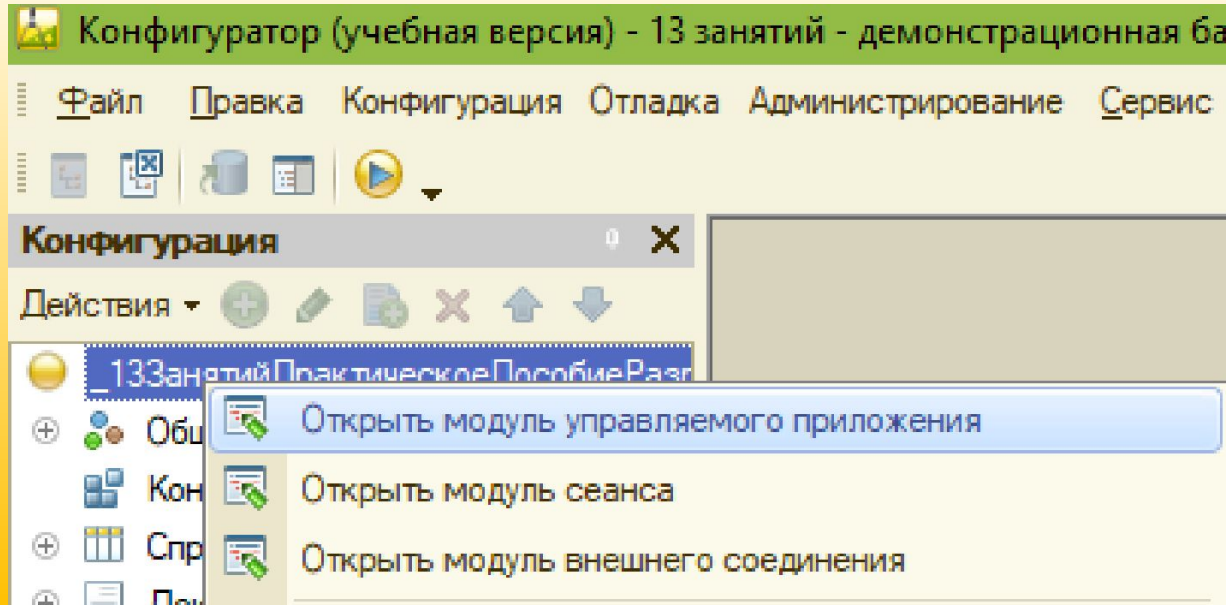
Виды программных модулей

1. **Модуль приложения (управляемого или обычного)**

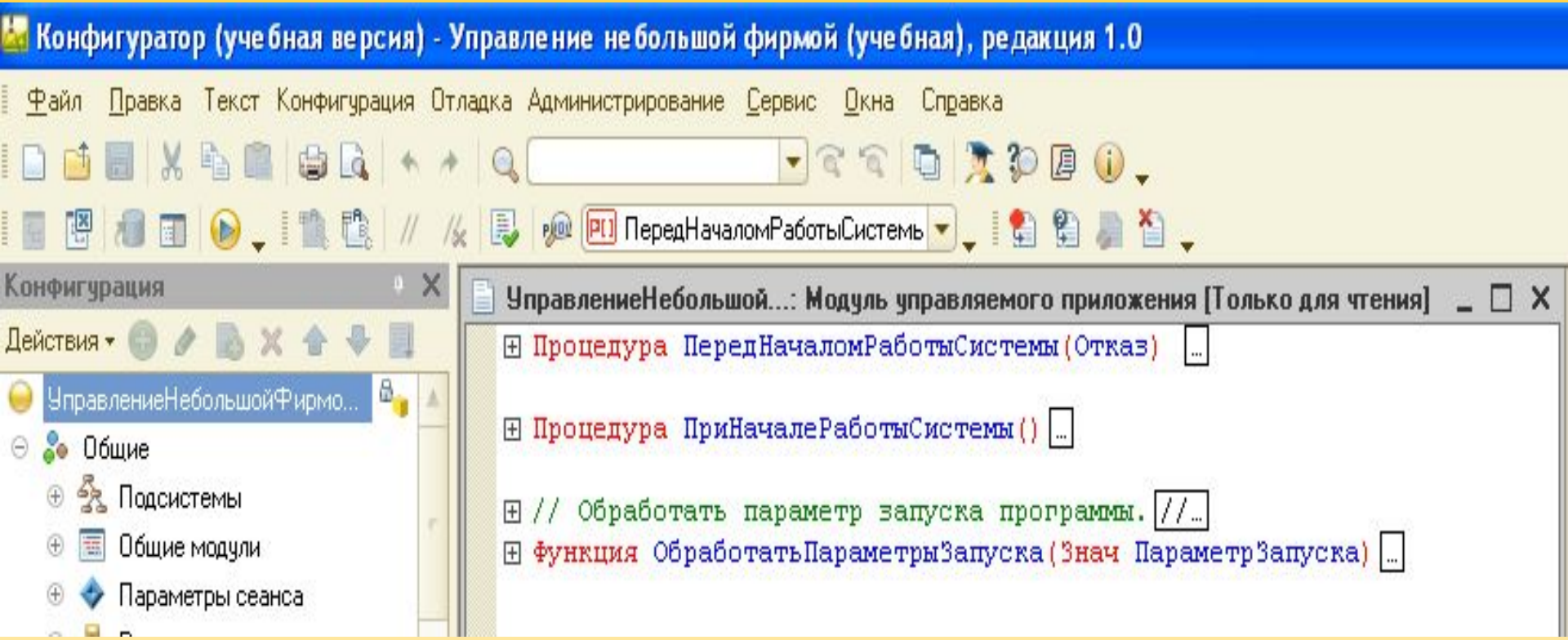
Модулем приложения называется модуль, который автоматически *выполняется в момент* ***загрузки конфигурации, при старте и окончании работы системы «1С:Предприятие».***

В модуле приложения описываются **процедуры (обработчики) событий, которые инициализируются при старте и окончании работы системы**.

Например, при запуске приложения можно загружать курсы валют из Интернета. При завершении приложения можно удостовериться у пользователя о его намерениях закончить работу.



Модуль приложения

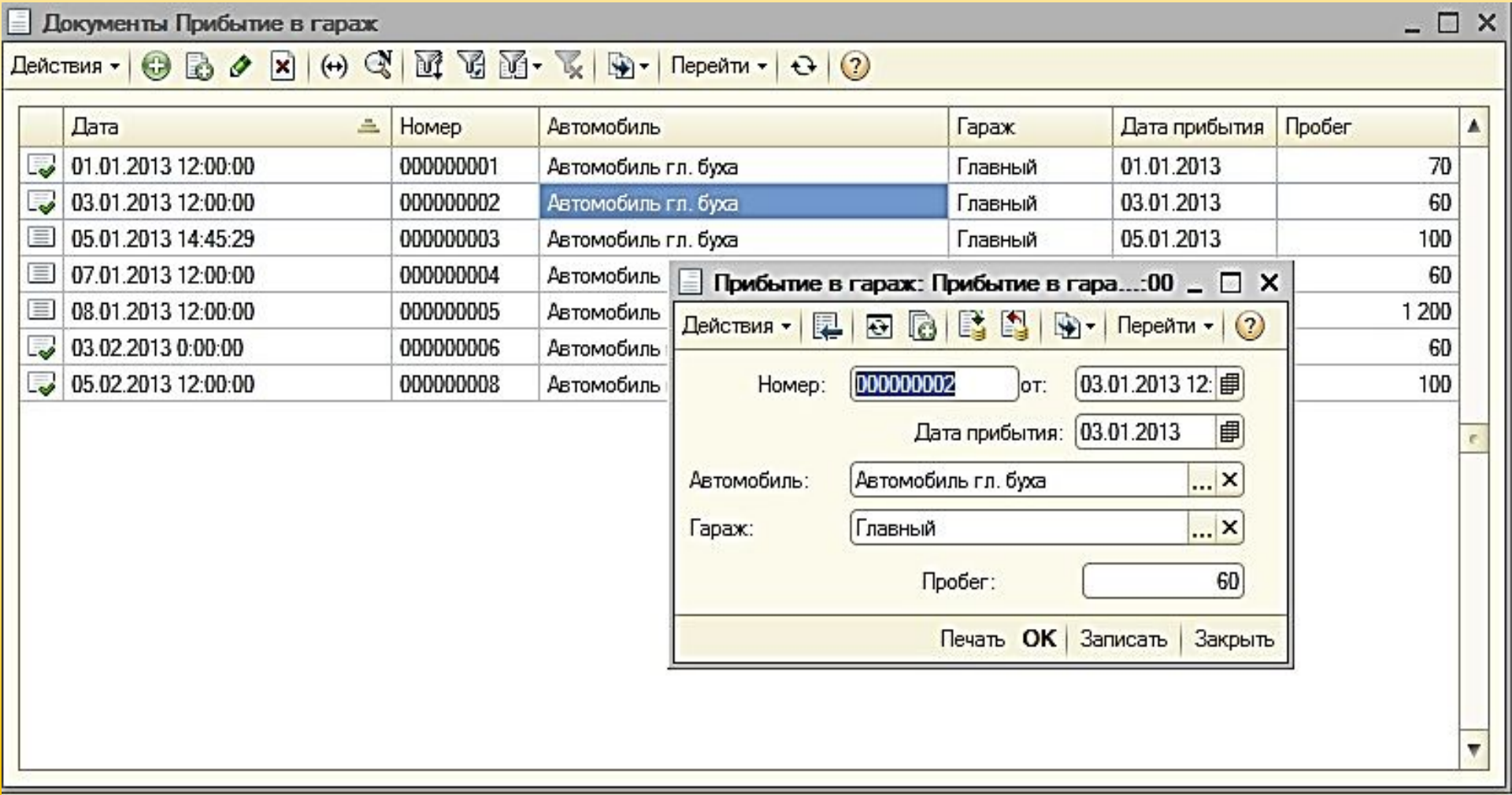


Модуль приложения

- В платформе 1С:Предприятие 8 существует два различных модуля приложения. **Это модуль Обычного приложения и модуль Управляемого приложения**. Они срабатывают при запуске различных клиентов. Так, **модуль Управляемого приложения** срабатывает при запуске ***веб-клиента, тонкого клиента и толстого клиента в режиме управляемого приложения***.
- А **модуль обычного приложения** срабатывает при ***запуске толстого клиента в режиме обычного приложения***.

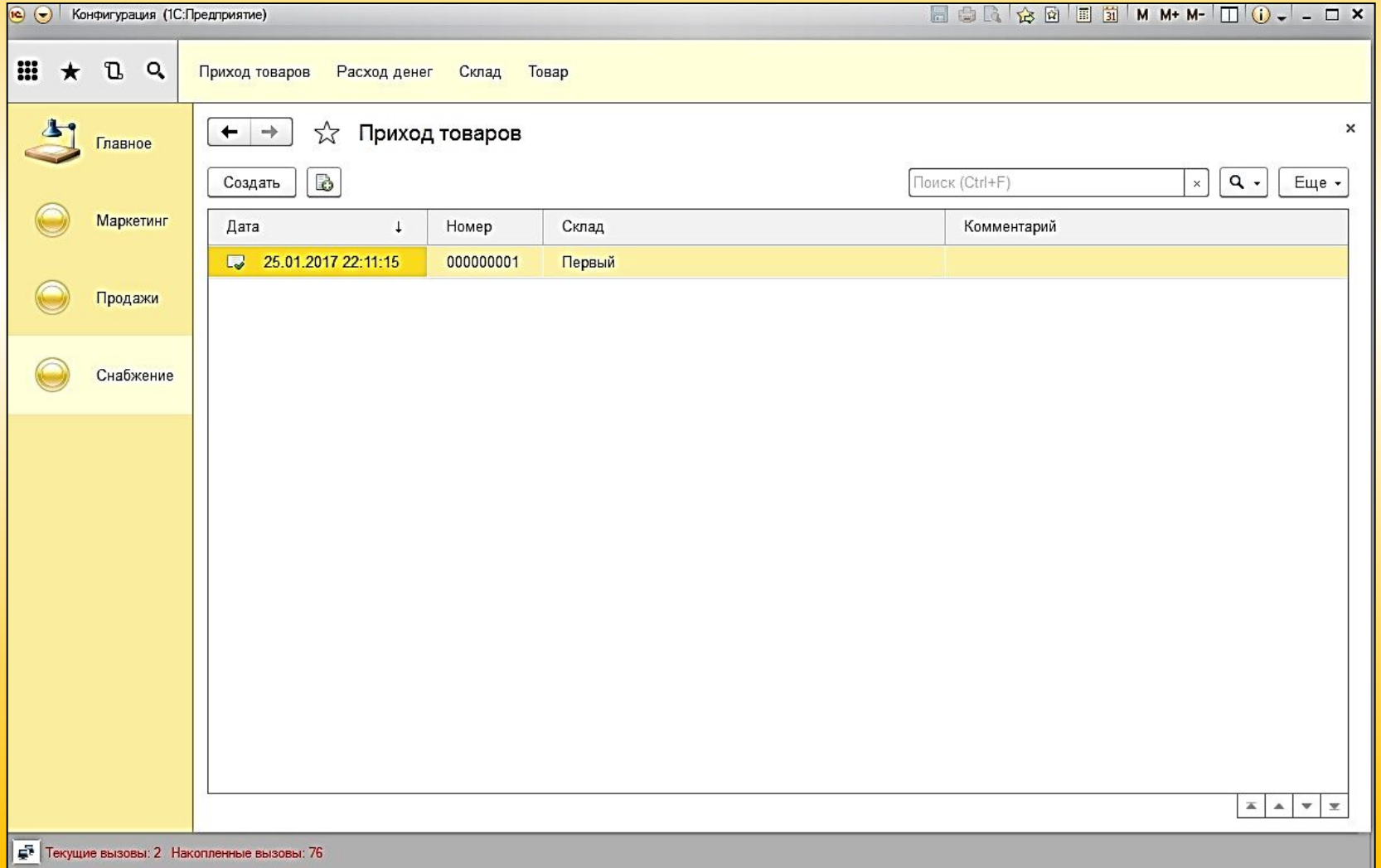
Обычное приложение

- В режиме обычного приложения используется обычные интерфейс и формы, которые применялись в платформах 8.0 и 8.1.



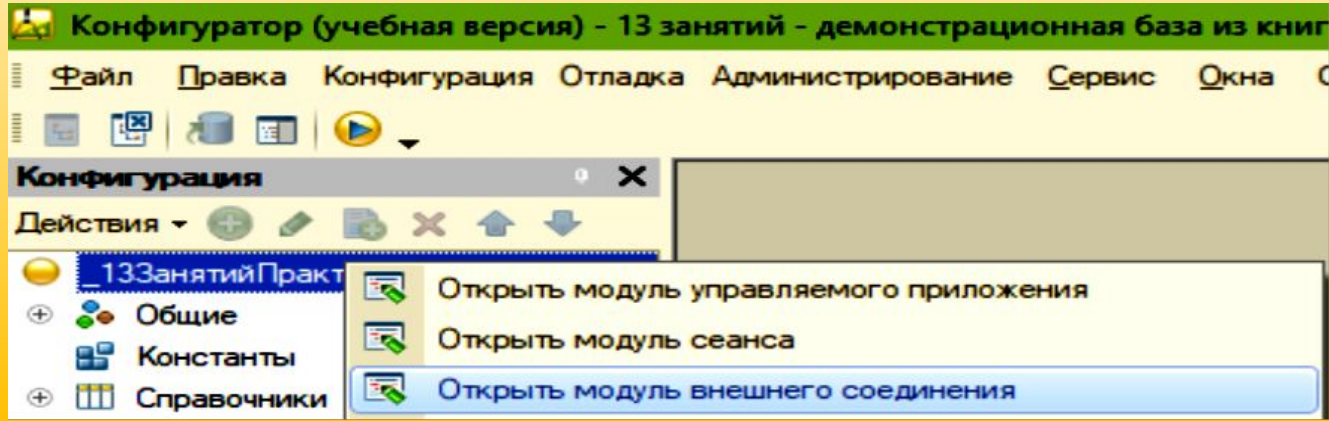
Управляемое приложение

- Основное отличие управляемого приложения от обычного — это использование **управляемого командного интерфейса и управляемых форм, которые в режиме 1С может настраивать пользователь..**



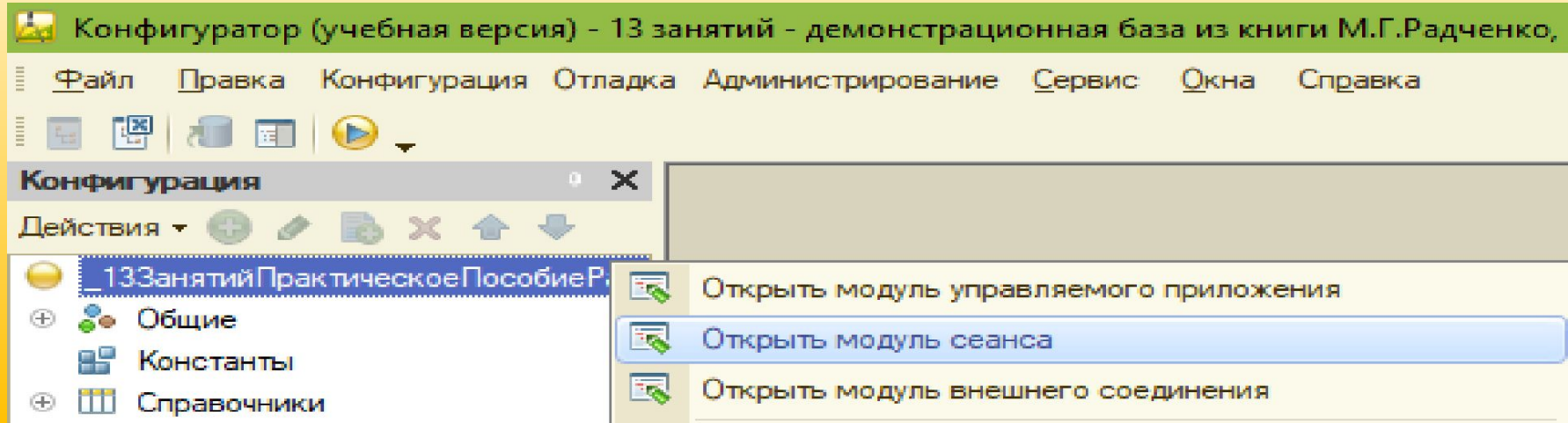
2. Модуль внешнего соединения

- В модуле внешнего соединения описываются **процедуры (обработчики) событий, которые инициализируются при старте и окончании работы системы в режиме внешнего соединения**
- **Внешнее соединение** - это *один из механизмов* ***интеграции с другими системами внешними по отношению к данной***.
- В этом режиме происходит программная ***работа с информационной базой и не происходит открытия окна приложения, т.е.*** процесс внешнего соединения – ***это процесс не интерактивный***.

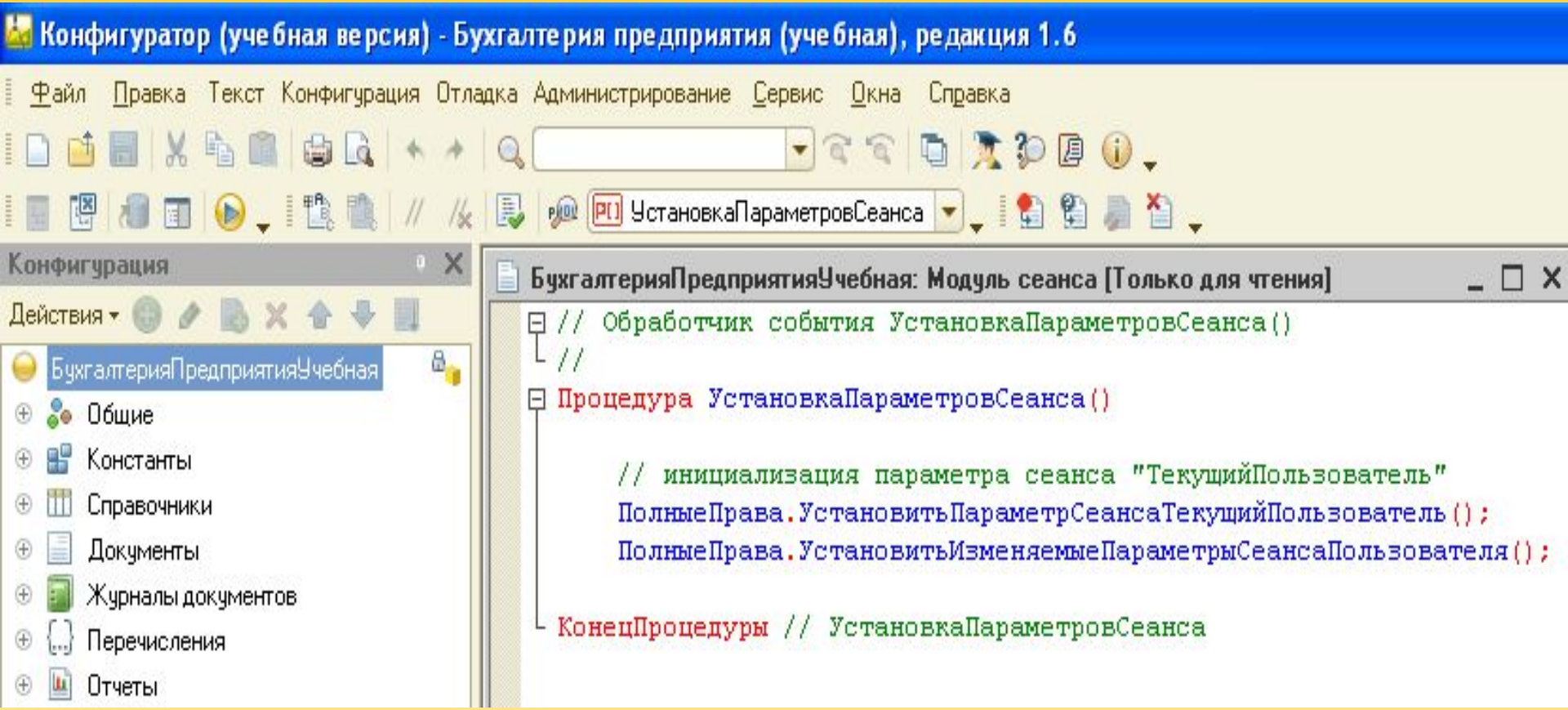


3. Модуль сеанса

Это узкоспециализированный модуль, предназначенный исключительно **для инициализации параметров сеанса**. Почему для этого необходимо было делать собственный модуль? Его использование обусловлено тем, что само приложение может запускаться в различных режимах (что приводит к выполнению либо *модуля управляемого, либо обычного приложения, либо модуля внешнего соединения*), а **инициализацию параметров сеанса необходимо производить вне зависимости от режима запуска**. Чтобы не писать один и тот же программный код во всех трех указанных модулях, и потребовался *дополнительный модуль (модуль сеанса), который выполняется вне зависимости от режима запуска приложения.*

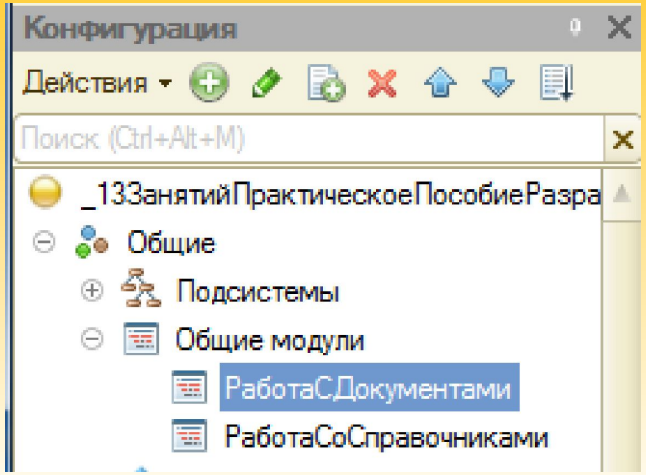


Модуль сеанса



В модуле сеанса существует одно единственное событие «УстановкаПараметровСеанса», которое выполняется самым первым, даже раньше события модуля приложения ПередНачаломРаботыСистемы.

Общие модули



Общие модули располагаются в отдельной ветке дерева метаданных. **Основным назначением общих модулей является содержание общих алгоритмов конфигурации, доступных из разных модулей.**

Общий модуль РаботаСДокументами: Модуль

```
Процедура РассчитатьСумму(СтрокаТабличнойЧасти) Экспорт

    СтрокаТабличнойЧасти.Сумма = СтрокаТабличнойЧасти.Количество *  СтрокаТабличнойЧасти.Цена;

КонецПроцедуры
```

Общий модуль РаботаСоСправочниками: Модуль

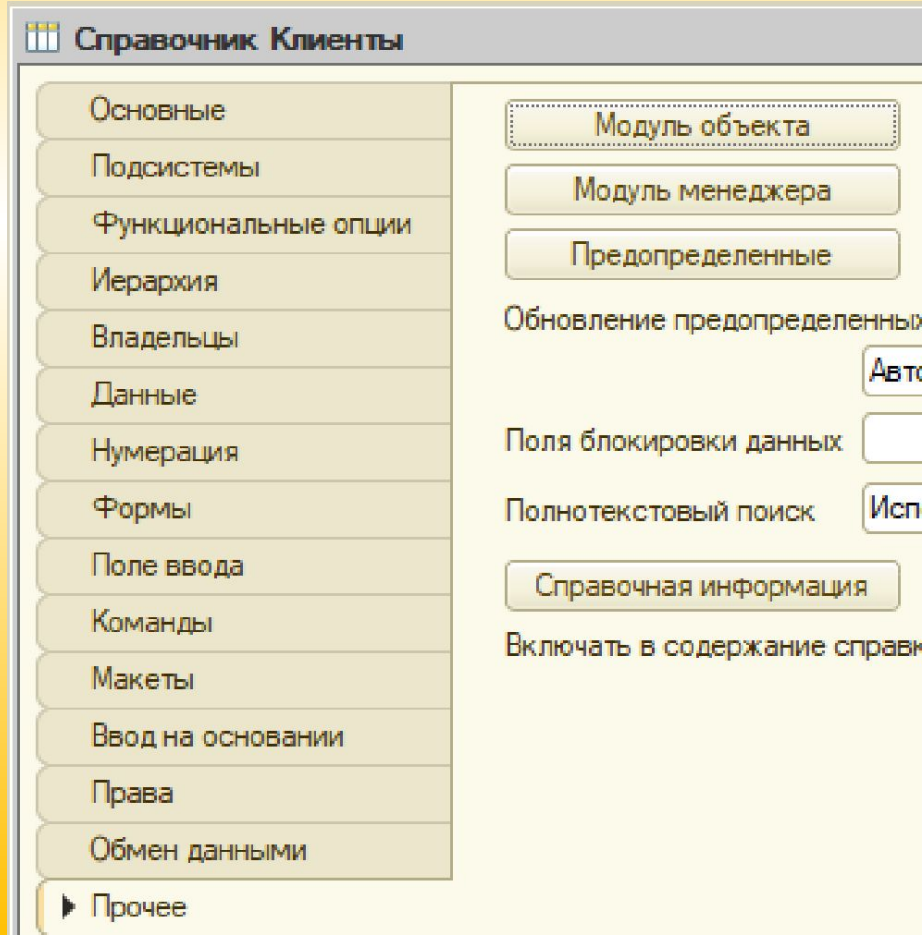
```
Функция РозничнаяЦена(АктуальнаяДата, ЭлементНоменклатуры) Экспорт

    // Создать вспомогательный объект Отбор
    Отбор = Новый Структура("Номенклатура", ЭлементНоменклатуры);

    // Получить актуальные значения ресурсов регистра
    ЗначенияРесурсов = РегистрыСведений.Цены.ПолучитьПоследнее(АктуальнаяДата, Отбор);
    Возврат ЗначенияРесурсов.Цена;

КонецФункции
```

Модули прикладных объектов

Набор прикладных объектов имеет собственные модули.

К таким объектам относятся:

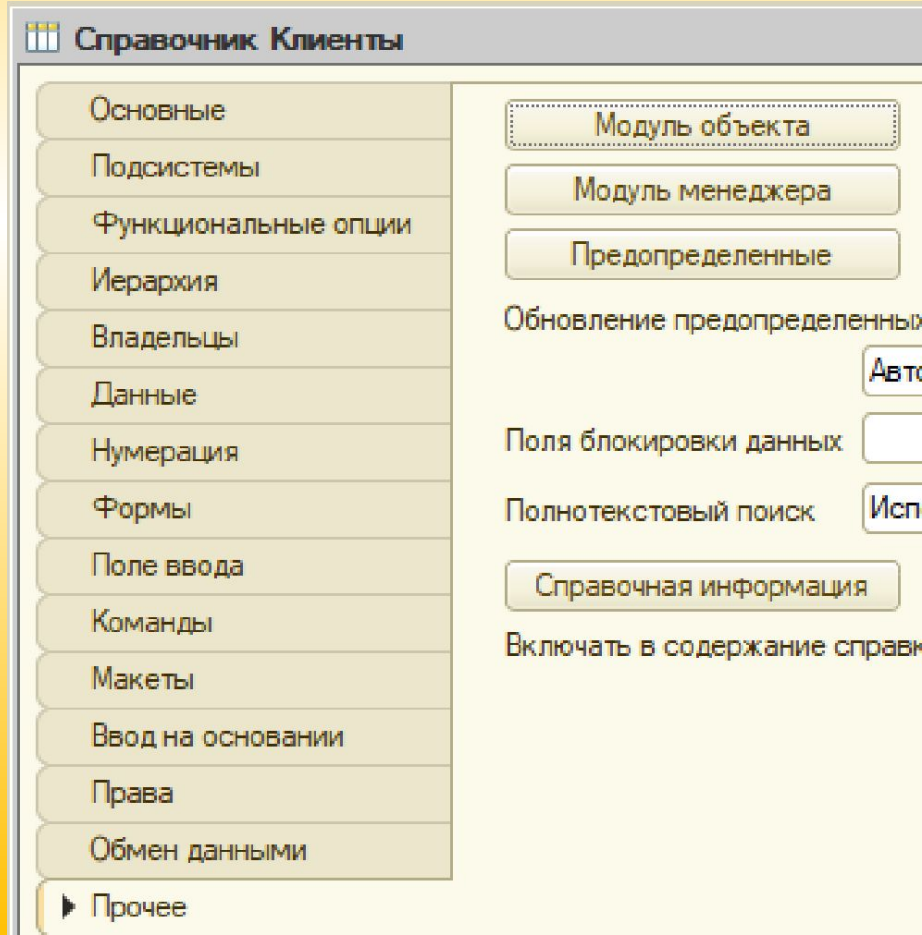
- ***Справочники,***
- ***Документы,***
- ***Отчеты,***
- ***Обработки,***
- ***Планы видов характеристик,***
- ***Планы счетов,***
- ***Планы видов расчетов,***
- ***Планы обмена,***
- ***Бизнес-процессы,***
- ***Задачи,***
- ***Регистры.***



Модуль объекта

Данный модуль предназначен для **обработки событий, непосредственно связанных с объектом**. Например, *события записи и удаления объектов, проверка заполнения реквизитов объекта, проведение документа* и т.д.

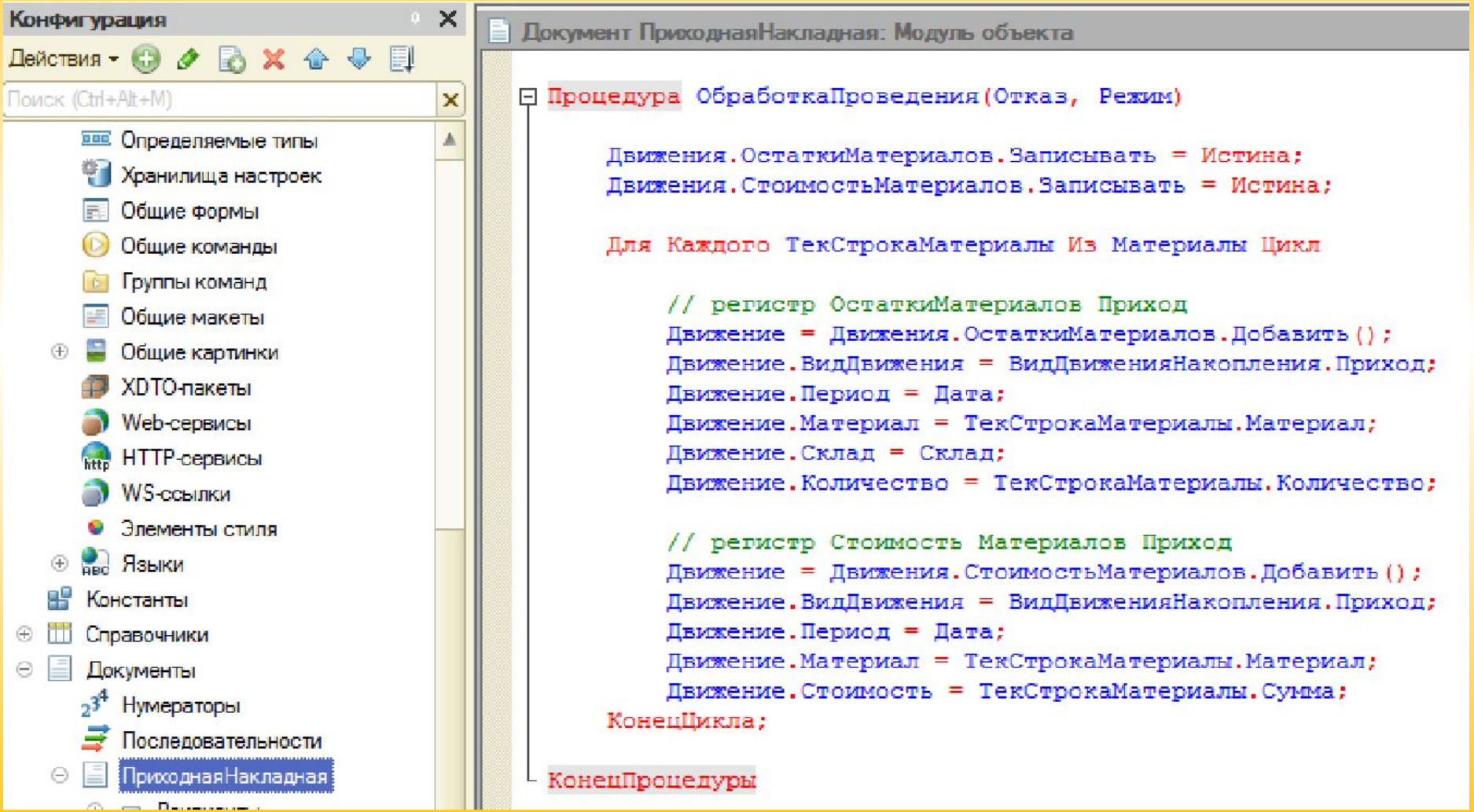
.



Модуль объекта

Модули располагаются в ветках конфигурации, в которых содержатся сами объекты, и **являются свойствами объектов**. Каждый объект имеет свой индивидуальный модуль.

Модуль объекта Приходная накладная



Модуль менеджера объекта

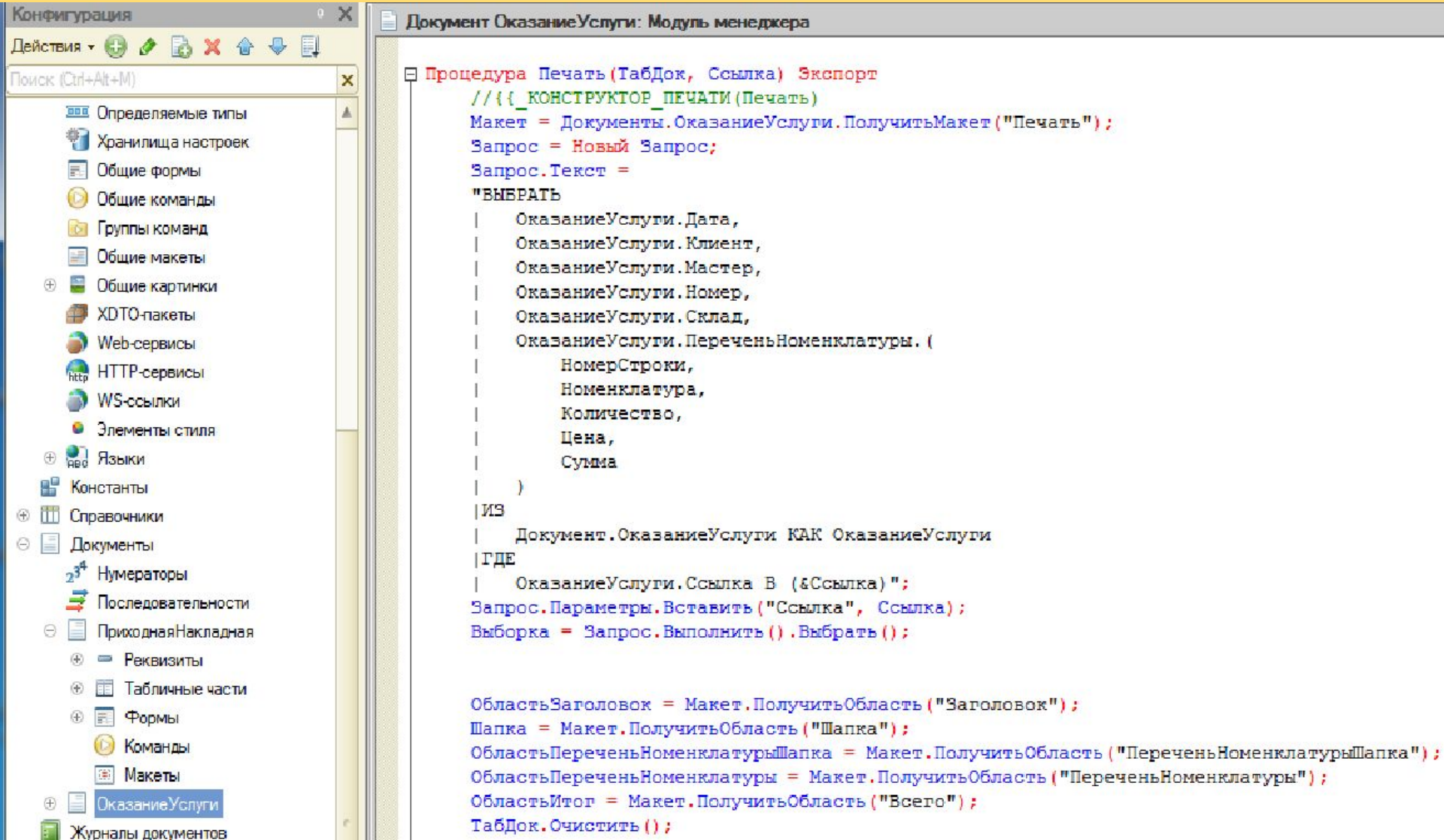
Каждый прикладной объект имеет менеджера, предназначенного ***для управления этим объектом как объектом конфигурации***.

С помощью менеджера можно ***создавать объекты, работать с формами и макетами***.

Модуль менеджера позволяет ***расширить функциональность менеджеров за счет введения процедур и функций на встроенном языке.*** Фактически это позволяет **описать методы для объекта конфигурации** (например, справочника), которые ***относятся не к конкретному экземпляру объекта базы данных, а к самому объекту конфигурации.***

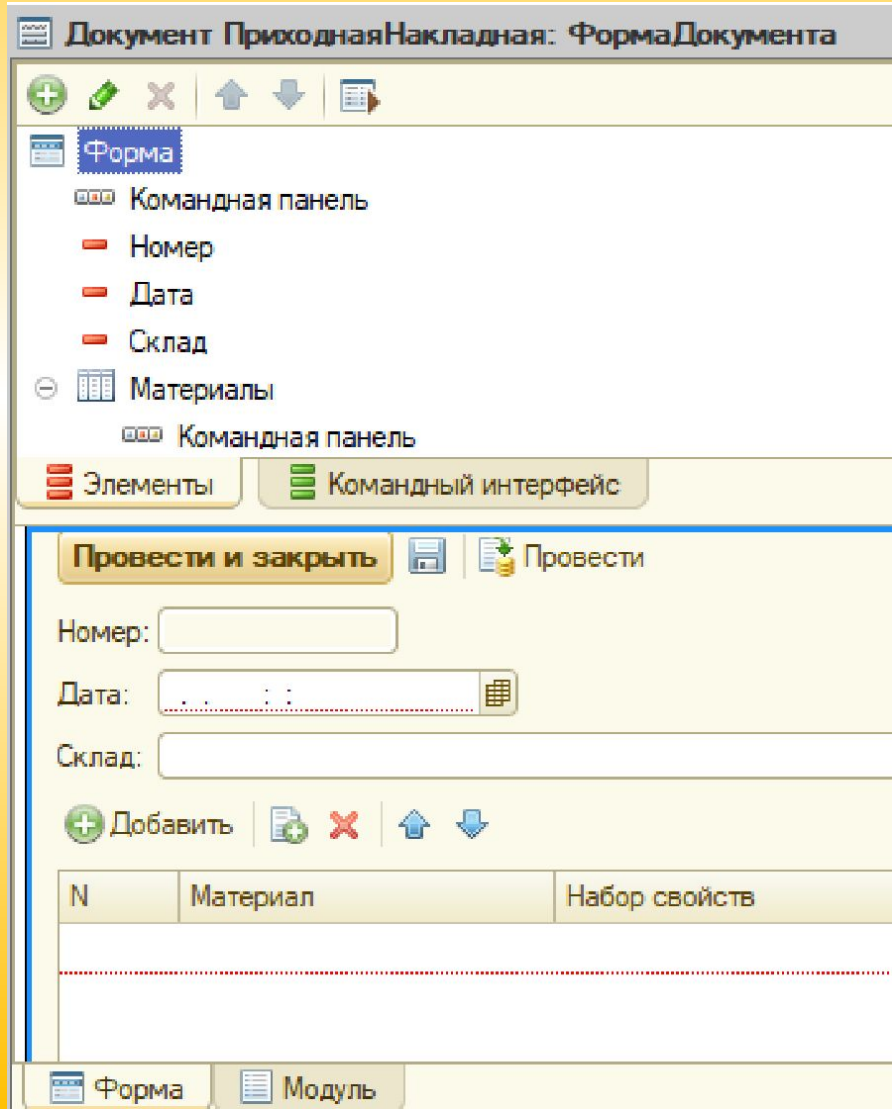
Примеры использования процедур и функций Модуля менеджеров объектов: *первоначальное заполнение отдельных реквизитов справочника или документа по определенным условиям, проверка заполнения реквизитов справочника или документа по определенным условиям* и т.д.

Модуль менеджера объекта документа Оказание услуги



Модуль формы

Модуль формы предназначен ***для обработки действий пользователя с данной формой*** (обработка события нажатия кнопки, изменения реквизита формы и т.д.). Так же существуют ***события связанные непосредственно с самой формой*** (например, ее открытие или закрытие).



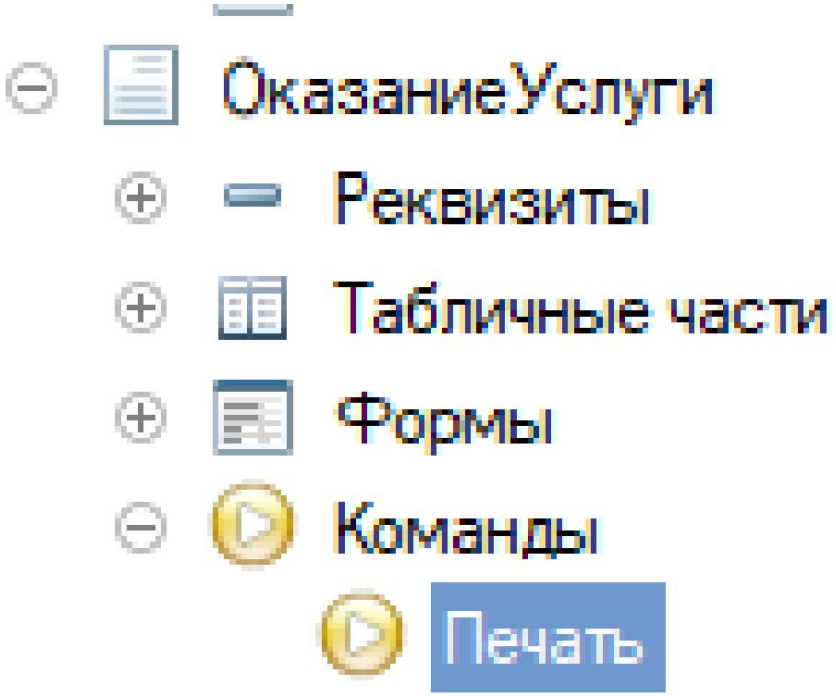
Модуль формы документа Оказание услуги

Документ ОказаниеУслуги: ФормаДокумента

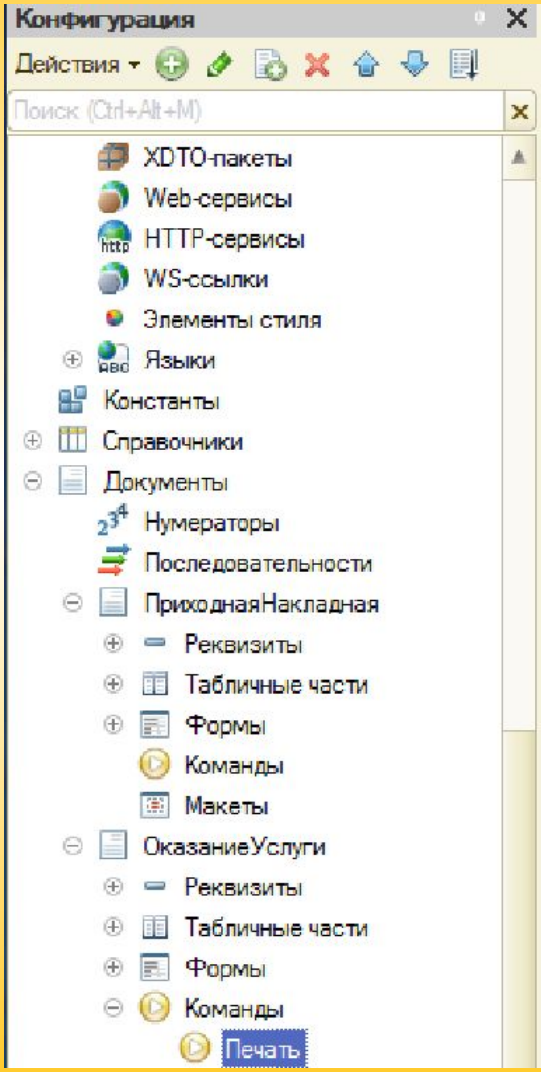
```
&НаКлиенте
Процедура ПереченьНоменклатурыКоличествоПриИзменении(Элемент) ...

&НаКлиенте
Процедура ПереченьНоменклатурыЦенаПриИзменении(Элемент) ...

&НаКлиенте
Процедура ПереченьНоменклатурыНоменклатураПриИзменении(Элемент)

    // Получить текущую строку табличной части
    СтрокаТабличнойЧасти = Элементы.ПереченьНоменклатуры.ТекущиеДанные;

    // Установить цену
    СтрокаТабличнойЧасти.Цена = РаботаСоСправочниками.РозничнаяЦена(Объект.Дата, СтрокаТабличнойЧасти.Номенкла

    // Пересчитать сумму строки
    РаботаСДокументами.РассчитатьСумму(СтрокаТабличнойЧасти);

        // Установить скидку
    СтрокаТабличнойЧасти.СкидкаНаМатериалы = СтрокаТабличнойЧасти.Сумма / 20;

    // Установить Итоговую сумму
    СтрокаТабличнойЧасти.ИтоговаяСумма = СтрокаТабличнойЧасти.Сумма - СтрокаТабличнойЧасти.СкидкаНаМатериалы;



КонецПроцедуры

&НаКлиенте
Процедура ПереченьНоменклатурыСуммаПриИзменении(Элемент) ...
```

Модуль команды

Модуль команды предназначен для того, чтобы описать в нем на встроенном языке те ***действия, которые должна выполнить система при вызове команды***.



Модуль команды Печать документа Оказание услуги



Команда Печать: Модуль команды

```
&НаКлиенте
Процедура ОбработкаКоманды(ПараметрКоманды, ПараметрыВыполненияКоманды)
    //{{_КОНСТРУКТОР_ПЕЧАТИ(Печать)
    ТабДок = Новый ТабличныйДокумент;
    Печать(ТабДок, ПараметрКоманды);

    ТабДок.ОтображатьСетку = Ложь;
    ТабДок.Защита = Ложь;
    ТабДок.ТолькоПросмотр = Ложь;
    ТабДок.ОтображатьЗаголовки = Ложь;
    ТабДок.Показать();
    //}}
КонецПроцедуры

&НаСервере
Процедура Печать(ТабДок, ПараметрКоманды)
    Документы.ОказаниеУслуги.Печать(ТабДок, ПараметрКоманды);
КонецПроцедуры
```

Команды – это объекты, подчиненные прикладным объектам или конфигурации в целом. У каждой команды есть модуль команды, в котором можно описать предопределенную процедуру ОбработкаКоманды() для выполнения этой команды.

Характеристики модулей 1С

Название модуля	Содержание	Выполнение	Расположение
Модуль приложения	**3 области программного модуля 1С**	выполняется на стороне **клиента**	в **корневом разделе конфигурации**
Модуль внешнего соединения	**3 области программного модуля 1С**	выполняется на стороне **сервера**	в **корневом разделе конфигурации**
Модуль сеанса	***1 область описания процедур и функций***	***выполняется на стороне сервера***	***в корневом разделе конфигурации***
Общий модуль	***1 область описания процедур и функций***	***выполняется на стороне сервера или клиента (зависит от настроек модуля)***	***в ветке дерева конфигурации «Общие» - «Общие модули»***
Модуль объекта	***3 области*** **программного модуля 1С**	***выполняется на стороне сервера***	***ссылка на вкладке Прочее описания объекта***
Модуль менеджера объекта	***1 область описания процедур и функций***	**выполняется на стороне сервера**	***ссылка на вкладке Прочее описания объекта***
Модуль формы	***3 области*** **программного модуля 1С**	***выполняется на стороне сервера и клиента***	***в форме объекта***
Модуль команды	***1 область описания процедур и функций***	**выполняется на стороне клиента**	***В ветви объекта конфигурации или Конфигурации в целом***

Контекст выполнения программного модуля

Каждый программный модуль связан с остальной частью конфигурации. Эта **связь** называется **контекстом выполнения модуля**. Контекст определяет ***«программное окружение», в котором исполняется код модуля, - набор доступных для модуля объектов, переменных, процедур и функций***.

Следует различать два вида контекста:

- *глобальный,*
- *локальный.*

Контекст выполнения программного модуля

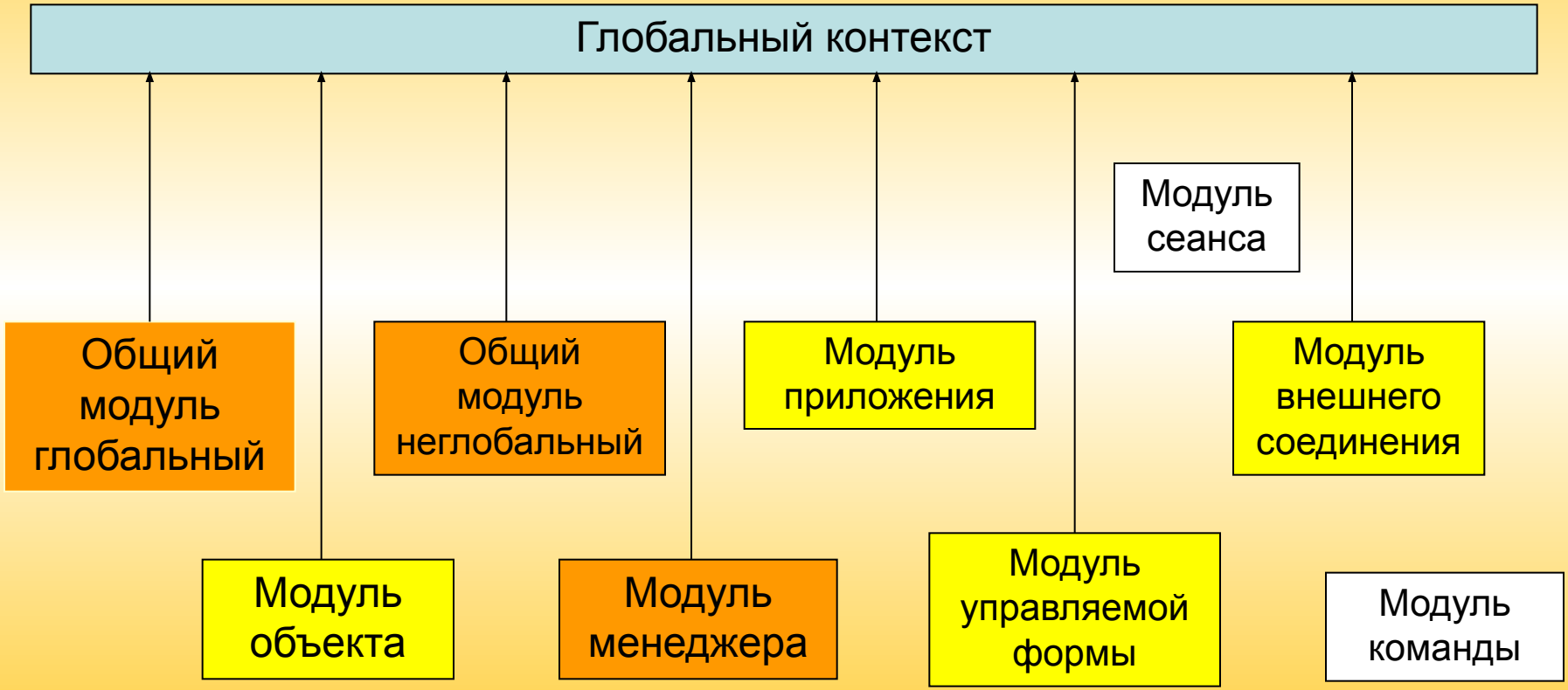
Глобальный контекст образуется:

- *значениями свойств и методов глобального контекста;*
- *системными перечислениями и системными наборами значений.*

Глобальный контекст виден всем программным модулям и определяет общую языковую среду конфигурации.

Почти все модули конфигурации (за исключением модуля сеанса и модуля команды) поставляют в глобальный контекст свои экспортируемые процедуры/функции.

Контексты прикладного решения



Экспортируют переменные, функции, процедуры

Экспортируют функции, процедуры

Контекст выполнения программного модуля

Локальный контекст модуля образуется тем *конкретным местом конфигурации задачи, для которого использован программный модуль*.

Локальный контекст виден только конкретному программному модулю и определяет для модуля набор непосредственно доступных модулю объектов, их свойств и методов.

Область использования переменной:

Область использования переменных зависит от места их определения в конфигурации. Существует три области, в которых можно объявить переменные:

1. В разделе определения переменных программного модуля управляемого приложения. Это глобальные переменные.
2. В разделе определения переменных модуля. Это переменные модуля.
3. В процедуре или функции. Это локальные переменные.

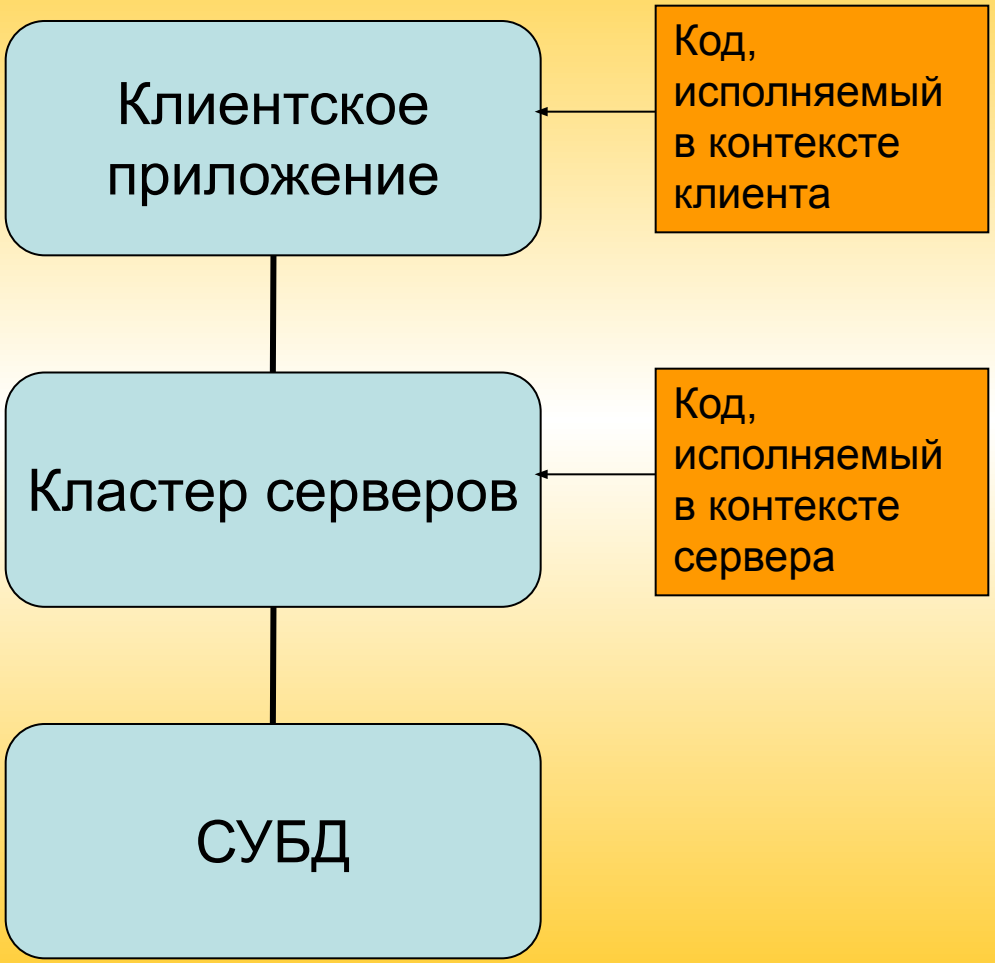
Глобальные переменные, объявленные с ключевым словом Экспорт, доступны для использования в исполняемых операторах, выражениях, в любой процедуре и функции любого клиентского программного модуля конфигурации.

Переменные модуля доступны для использования в исполняемых операторах, выражениях, в любой процедуре и функции того программного модуля, в пределах которого они объявлены. Если они объявлены с ключевым словом Экспорт, то они доступны из других модулей через контекст модуля, в котором они объявлены.

Локальные переменные доступны в пределах той процедуры или функции, в которой они объявлены.

Контекст выполнения модулей

- Контекст выполнения модулей определяет *программную среду, в которой выполняется модуль.*



Контекст выполнения модулей

Наличие серверного и клиентского контекста исполнения модулей определяет следующие особенности:

В контексте клиента и в контексте сервера доступны разные свойства, методы и объекты встроенного языка. **Все действия, связанные с доступом к данным (их чтение и запись), возможны только на сервере, а отображение этих данных пользователю и другие интерактивные действия возможны только на клиенте.** Поэтому ***клиентские процедуры в модулях в явном виде отделяются от серверных, и в них используется ограниченный состав объектной модели встроенного языка.***