

Лабораторная работа № 2. Основные виды объектов конфигурации.

Цель лабораторной работы № 2: Изучение основных видов и объектов конфигурирования на платформе 1С Предприятие.

Задачи работы – приобретение навыков:

- Создание элементов конфигурации Константы;
- Создание форм констант;
- Изучение механизмов создания печатных форм справочников;
- Изучение объекта конфигурации «Перечисления»;
- Изучение процесса редактирования форм.

1. Теоретические основы конфигурирования в системе 1С Предприятие 8

1.1 Классификация объектов конфигурации

Все объекты конфигурации образуют несколько основных видов. Их можно разделить на три основные группы:

1. **Общие объекты.** Группа вспомогательных объектов конфигурации, с помощью которых осуществляется создание конфигурации, механизмов взаимодействия пользователей с учетными данными.

2. **Прикладные объекты.** Их перечень можно увидеть на первом уровне дерева метаданных (исключая группу «Общие»).

3. **Подчиненные объекты.** К ним относятся «Реквизиты», «Табличные части» и т.д.

К прикладным объектам относятся:

1. **Константы.** Предназначены для хранения постоянных, условно-постоянных величин.

2. **Справочники.** Списки однородных элементов данных. Используются для хранения нормативно-справочной информации.

3. **Документы.** Служат для ввода информации о совершаемых операциях в системе.

4. **Журналы документов.** Служат для отображения списков документов различного вида.

5. **Перечисления.** Списки значений, задаваемых на этапе конфигурирования.

6. **Отчеты.** Средство получения выходной информации.

7. **Обработки.** Используются для выполнения различных действий над информационной базой.

8. **Планы счетов.** Совокупность синтетических счетов.

9. **Планы видов характеристик.** Предназначены для описания множеств однотипных объектов аналитического учета.

10. **Планы видов расчета.** Предназначены для описания множеств однотипных объектов механизмов расчета.

11. **Регистры сведений.** Служат для хранения информации, состав которой развернут по определенной комбинации значений и, при необходимости, развернут во времени.

12. **Регистры накопления.** Служат для накопления информации в разрезе измерений с возможностью получения остатков или оборотов числовых величин (или остатков и оборотов).

13. **Регистры расчетов.** Служат для накопления информации о периодических расчетах.

14. **Регистры бухгалтерии.** Используются для отражения в бухгалтерском учете информации о хозяйственных операциях.

15. **Бизнес-процессы.** Используются для реализации «процессного» принципа работы. Данный принцип позволяет автоматизировать процесс прохождения и контроля цепочек событий, операций.

16. **Задачи.** Совместно с бизнес-процессами реализуют процессный принцип. Они позволяют вести учет заданий по исполнителям и служат отражением продвижения бизнес-процессов по точкам маршрута.

17. **Внешние источники.** Позволяют организовать более удобную работу с внешними источниками данных (в основе лежит механизм ODBC).

К подчиненным объектам относятся:

1. **Реквизиты** – дополнительная информация об объекте, доступная только в пределах этого объекта. С помощью реквизитов можно определить дополнительные свойства объекта.

2. **Табличные части** – наборы дополнительной информации об объекте, представленной в виде таблиц.

3. **Реквизиты табличных частей** – состав табличной части объекта, доступны только в пределах табличной части объекта.

4. **Формы** – используются для ввода, просмотра и редактирования информации.

5. **Команды** – используются для реализации каких-либо действий, принадлежащих объекту.

6. **Макеты** – предназначены для формирования печатных форм объекта.

7. **Графы** – графы журнала документов.

8. **Измерения** – для регистров это объекты конфигурации, в разрезе которых указываются данные в регистре.

9. **Ресурсы** – данные, учитываемые в регистре.

В «1С:Предприятие» используется **принцип учета «от документа»**: *деятельность организации разбивается на элементарные операции. Под каждую операцию создается объект «Документ». Документами в систему вносится первичная информация о совершенной операции. При заполнении документов используется дополнительная справочная информация. Информация из документов попадает в учетные объекты – регистры. Данные в регистрах могут быть откорректированы различными регламентами (регламентными документами, обработками).*

1.2 Типы данных

Большинство объектов, из которых строится программный комплекс, имеют свойство «Тип данных».

Различают три основных группы типов данных:

1. **Примитивные типы** (в их состав входят базовые типы данных).
2. **Типы данных, зависимые от метаданных**, появившиеся после определения в конфигурации объектов конфигурации.
3. **«Другие» типы**, не относящиеся к примитивным и «добавляемым», но поддержка которых во встроенном языке есть изначально.

К примитивным типам данных относятся:

1. <Число> (десятичное число).
2. <Строка> (строка фиксированной, переменной или неограниченной длины).
3. <Дата> (дата, время, дата+время).
4. <Булево> (истина или ложь)
5. <Тип>.
6. <Неопределенно>.
7. <Null>.

1.3 Универсальные коллекции значений

«Другие» типы, не относящиеся к примитивным и «добавляемым», но поддержка которых во встроенном языке есть изначально, иногда являются коллекциями (их можно «обойти» как по индексу, так и с помощью специального вида цикла «Для Каждого Из»). Часть из этих типов входит в так называемые «Универсальные коллекции значений». Универсальные коллекции значений предназначены для хранения временных наборов данных в течение сеанса работы пользователя. Они не являются объектами информационной базы и служат для вспомогательного сбора, группировки, анализа и обработки информации.

Массив.

Объекты данного типа представляют собой совокупность значений любого типа, в том числе и типа «массив», что, в частности, позволяет организовывать многомерные массивы.

Объект создается из программного кода с использованием конструктора «Новый».

Массив = НовыйМассив (кол-во элем 1,...N);

Пример кода:

```
Массив = НовыйМассив;  
Массив.Добавить(«Первый»);  
Массив.Добавить(2);  
//так далее
```

Структура.

Структура представляет собой динамический набор данных – коллекцию значений, каждый элемент которой состоит из пары «Ключ» и «Значение». Ключи структуры уникальны, и поэтому ими можно идентифицировать значения. Ключ структуры должен быть строковым и отвечать требованиям к именам переменных. К значениям структуры можно обращаться как к свойствам объекта, при этом ключ используется как имя свойства.

СтруктураОтбора = Новый Структура(«Ключи»,Значения);

Пример кода:

Отбор = Новый Структура(«Валюта, Контрагент», Валюта, Контрагент);

Другой вариант создания структуры:

СтруктураОтбора = Новый Структура;

СтруктураОтбора.Вставить(«Валюта»,Валюта);

СтруктураОтбора.Вставить(«Контрагент»,Контрагент);

Соответствие.

Соответствие представляет собой динамический набор данных – коллекцию значений, каждый элемент которой состоит из пары «Ключ» и «Значение». Ключи соответствия уникальны, и поэтому ими можно идентифицировать значения. В отличие от ключа структуры, ключи соответствия могут быть произвольных типов. Рекомендуется, чтобы в качестве ключа выступало значение неизменяемого типа или другого типа, значение которого может только присваиваться, но не может менять свое содержимое.

Соотв = Новый Соответствие();

Список значений.

Список значений это объект, позволяющий строить динамические наборы значений и манипулировать ими. Может быть наполнен значениями любых типов. Условно список значений можно представить как таблицу из четырех колонок: пометка, значение, представление, картинка. Каждое из значений характеризуется позицией в списке (индексом).

СПЗ = Новый СписокЗначений

Таблица значений.

Таблица значений – объект, позволяющий строить динамические наборы значений и манипулировать ими. Он может быть наполнен

значениями различных типов. Может иметь любое количество колонок и быть связанным с элементом «табличное поле».

ТЗ = Новый ТаблицаЗначений

Пример кода:

```
ТаблицаЗначений = Новый ТаблицаЗначений;  
ТаблицаЗначений.Колонки.Добавить(«Количество», «Количество  
товара»);  
СтрокаТаблицыЗначений = ТаблицаЗначений.Добавить();  
СтрокаТаблицыЗначений.Количество = 11;
```

Дерево значений.

Объект, похожий на таблицу значений. Но, в отличие от нее, строки дерева значений могут образовывать иерархические структуры: каждая строка дерева может иметь набор подчиненных строк и т.д.

ДЗ = Новый ДеревоЗначений();

1.4 Встроенный язык системы

Необходимость наличия встроенного языка определена концепцией настраиваемости системы. Язык является предметно-ориентированным. Он поддерживает специализированные типы данных предметной области, определяемые конфигурацией системы. Работа с этими типами данных в языке организована с использованием объектной техники.

Язык поддерживает конструкции, позволяющие определять переменные, процедуры, функции. Операторы отделяются друг от друга символом «;».

Встроенный язык не чувствителен к регистру, допускается двузязычное описание конструкций (Если, if). Рекомендуется писать на языке типовых конфигураций.

Перем ИмяПеременной;

Процедура ИмяПроцедуры(ИмяПараметра1, ИмяПараметра2, ...)

//текст комментария

//тело процедуры

КонецПроцедуры

Функция ИмяФункции(ИмяПараметра1, ИмяПараметра2, ...)

//тело функции

Возврат(ВозвращаемоеЗначение);

КонецФункции

Имя переменной, процедуры, функции может состоять из букв, цифр и символов подчеркивания. Начинаться имя должно либо с буквы, либо с символа подчеркивания.

Порядок описания процедур, функций между собой значения не имеет. Как и в любом другом языке существуют конструкции, реализующие ветвление, циклы.

Если Условие Тогда

//код

ИначеЕсли Условие Тогда

//код

Иначе

//код

КонецЕсли;

Для ПеременнаяСчетчик = НачальноЗначение По Конечное Цикл

//тело цикла

КонецЦикла;

Для Каждого ПеременнаяЦикла Из ИмяКоллекции Цикл

//тело цикла

КонецЦикла;

Пока УсловиеЦикла Цикл

//тело цикла

КонецЦикла;

Часто во встроенном языке придется иметь дело с ними объектами сущностями (с объектами, имеющими набор свойств, методов). Для обращения к свойству объекта можно использовать два подхода:

Наим = Спр.Наименование;

Наим = Спр[«Наименование»];

Вызов методов объектов производится «через точку».

Спр.Печать();

Допускаются следующие конструкции:

Док.Контрагент.ПолучитьОбъект().ПечатьКарточкиКлиента();

Платформа сочетает в себе визуальные и языковые средства конфигурирования. Использование встроенного языка в системе имеет

событийно-зависимую ориентацию, т.е. языковые модули используются в конкретных местах для обработки отдельных алгоритмов, настраиваемых в процессе конфигурации. Программный код всегда помещается в модули.

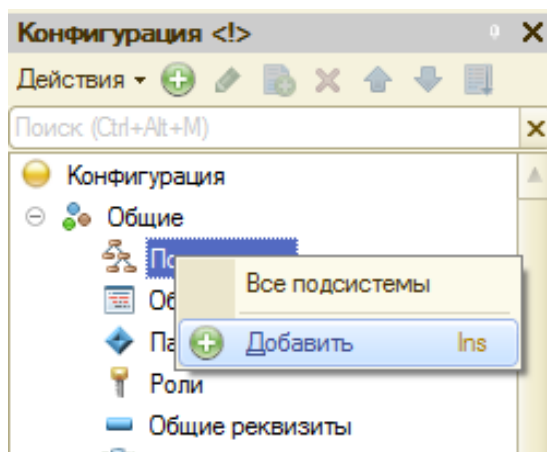
Место размещения конкретного программного модуля предоставляется конфигуратором в тех точках конфигурации, которые требуют описания специфических алгоритмов функционирования. Эти алгоритмы следует оформлять в виде процедур или функций, которые могут быть вызваны самой системой в заранее предусмотренных ситуациях.

2. Выполнение лабораторной работы.

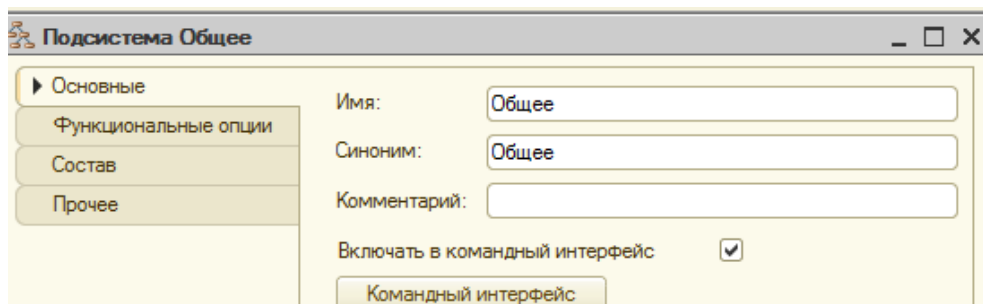
2.1 Подсистемы

Структура подсистем определяет структуру функциональности прикладного решения. Структура подсистем определяет, каким образом пользователь будет осуществлять «навигацию» по функциональности предлагаемого решения.

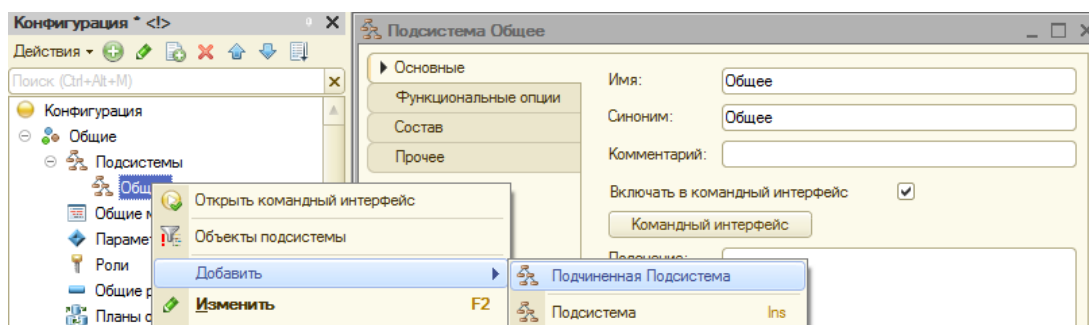
Для создания подсистем нужно зайти внутрь ветви «Общие» и выполнить команду «Добавить» контекстного меню ветви «Подсистемы».



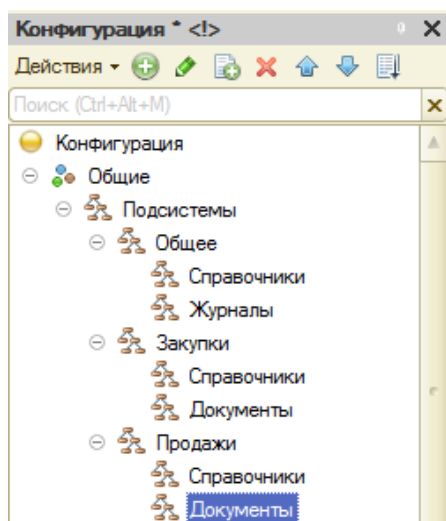
Обратите внимание на флаг «включать в командный интерфейс». Подсистемы со снятым флагом не влияют на структуру функциональности программного комплекса.



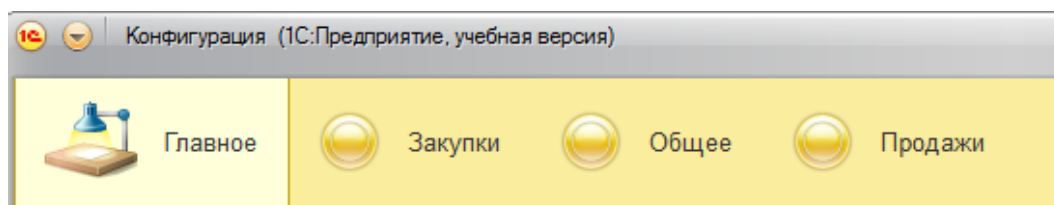
К подсистемам первого уровня можно добавить подчиненные подсистемы и т.д.



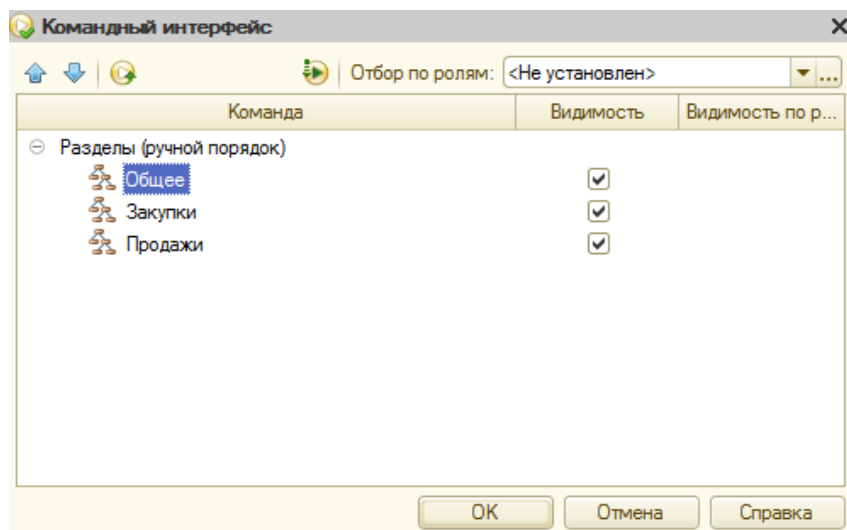
Количество уровней подсистем не ограничено.
Определите подсистемы таким же образом, как на рисунке.



Подсистемы первого уровня определяют структуру «панели разделов». Это можно увидеть, запустив «1С:Предприятие».



Изменить порядок следования подсистем можно при помощи команды контекстного меню корня дерева объекта конфигурации «Открыть командный интерфейс конфигурации». Либо в палитре свойств найти одноименные свойства и воспользоваться гиперссылкой. В открывшемся окне можно изменить порядок следования разделов.



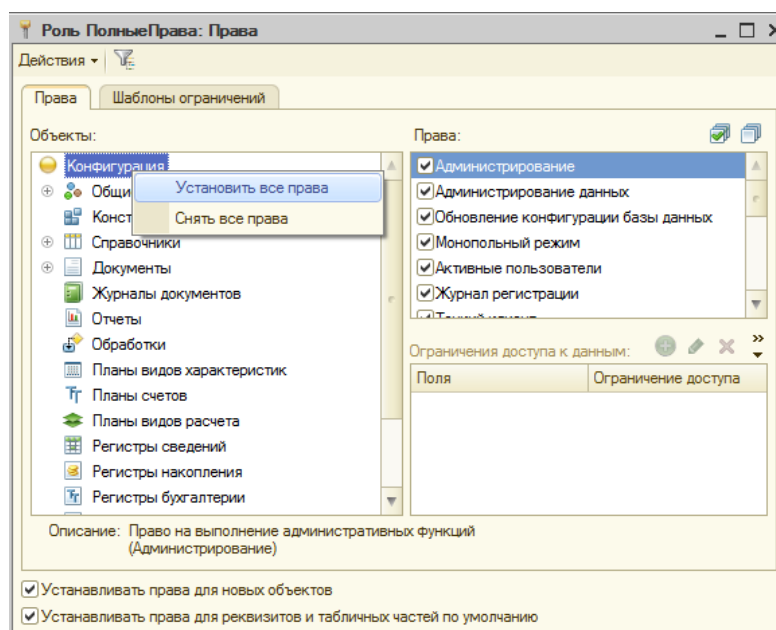
2.2 Роли

С помощью ролей в дальнейшем будет определяться доступность какой-либо функциональности для определенной группы пользователей конфигурации (также роли влияют на интерфейс системы).

Состав ролей следующий:

1. **ПолныеПрава** (должны соответствовать своему наименованию).
2. **Продажи** (предполагается, что у данной роли будет отсутствовать доступ к функционированию системы, связанной с проведением закупок).
3. **Закупки** (предполагается, что у этой роли будет отсутствовать доступ к функциональности системы, связанной с проведением продаж).

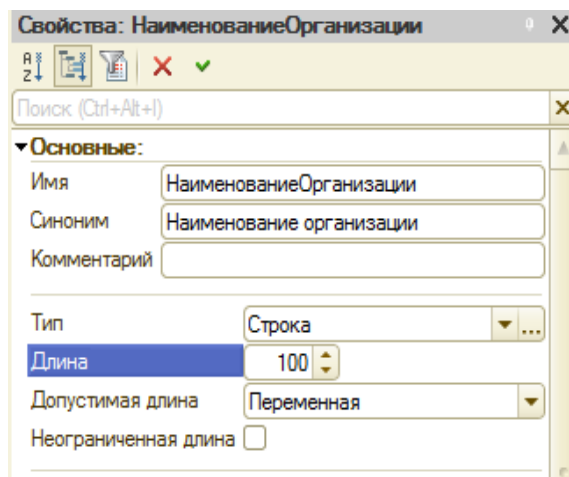
Для первой роли установим все права. Также следует установить флаг «Устанавливать права для новых объектов».



2.2 Константы

В любой организации существует набор «значений», которые не меняются довольно длительное время. К ним можно отнести название фирмы, юридический адрес, фамилии ответственных лиц и т.д. Для хранения таких значений идеально подходят константы.

Создадим константу «НаименованиеОрганизации». Для этого сделаем щелчок правой клавишей мыши на ветке «Константы» и выберем пункт «Добавить». В открывшемся окне свойств заполним необходимые реквизиты.



Создав константу мы определили возможность хранение в базе значений указанного типа.

Разработчик прикладного решения может работать с константой следующим образом:

//на чтение

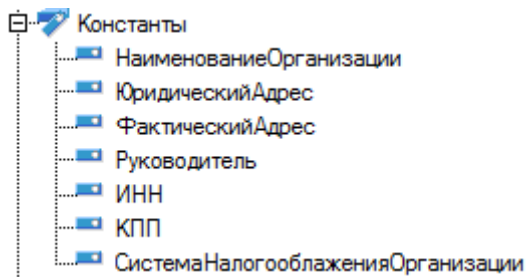
МояПеременная=Константы.НаименованиеОрганизации.Получить();

//установка значения

НовоеЗначение=«Новая организация»;

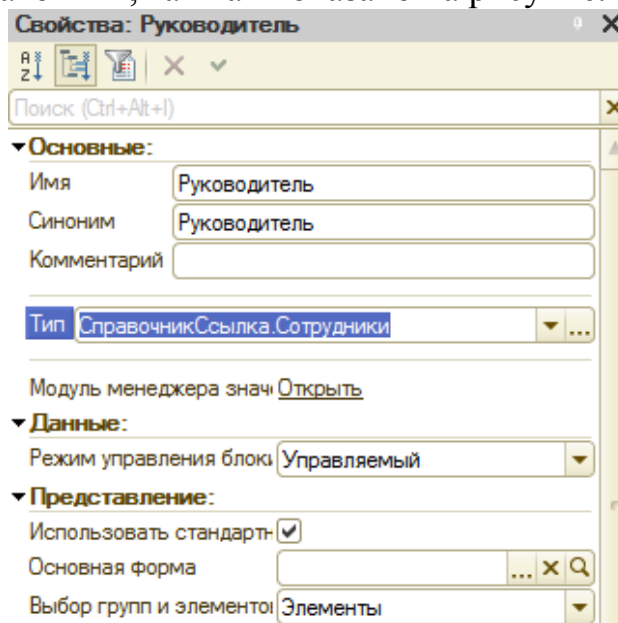
Константы.НаименованиеОрганизации.Установить(НовоеЗначение);

Далее необходимо создать константы, показанные на следующем рисунке



При создании констант «Юридический адрес» и «Фактический адрес» следует поставить Тип - Строка, и поставить галочку «неограниченная длина», что позволит нам вводить данные, неограниченной длины.

При создании константы «Руководитель» следует указать тип – соответствующий справочник, так как показано на рисунке.



Свойства: Руководитель

Поиск (Ctrl+Alt+I)

▼ Основные:

Имя: Руководитель

Синоним: Руководитель

Комментарий:

Тип: СправочникСсылка.Сотрудники

Модуль менеджера знач: Открыть

▼ Данные:

Режим управления блоком: Управляемый

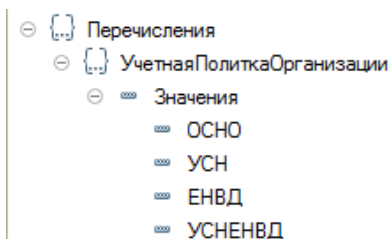
▼ Представление:

Использовать стандарт: ☒

Основная форма: ... X Q

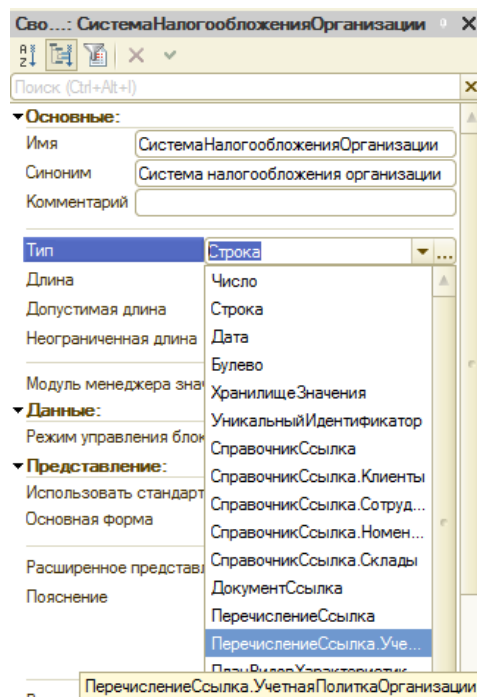
Выбор групп и элементов: Элементы

Для создания константы «Система налогообложения организации» необходимо прибегнуть к помощи объекта конфигурации **Перечисление** (Списки значений, задаваемых на этапе конфигурирования) т.е. пропишем список значений на вкладке «Данные», который будет выбираться из списка. Создадим перечисление «Учетная Политика Организации» как показано на рисунке:



В значении перечисления «УСНЕНВД» выставить синоним «УСН + ЕНВД».

После этого в свойстве константы «Система налогообложения организации» тип – выбираем созданное нами перечисление.



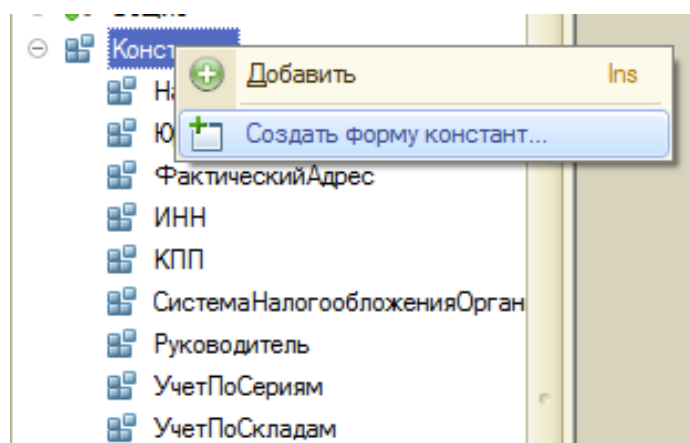
Задание: создайте дополнительно две константы:

1. **Имя: «УчетПоСериям», тип: «Булево».**
2. **Имя: «УчетПоСкладам», тип: «Булево».**

Формы констант

На этом работа с константами не завершилась. Мало того, что они были созданы, необходимо создать форму для их просмотра и редактирования.

Для ее создания необходимо щелкнуть правой клавишей мыши на ветке «Константы» и выбрать пункт контекстного меню: «Создать форму констант».



Результатом выполнения команды будет запуск конструктора формы. Следует обратить внимание на то, что повторно запустить конструктор для уже созданной формы не получится.

Общие принципы работы с конструктором форм одинаковы и не зависят от того, для какого объекта создается форма. Это несколько этапов:

1. Выбор типа формы. Выбор типа формы напрямую влияет на ее функциональность, предоставляемой по умолчанию
2. Определение имени формы.
3. Контроль значения флага «Использовать стандартные команды». По умолчанию флаг установлен.
4. После нажатия на кнопку «Далее» можно определить состав отображаемых объектов.

Конструктор общих форм

Выберите тип формы:

- ☐ Произвольная форма
- ☒ Форма констант
- ☐ Форма отчета
- ☐ Форма настроек отчета
- ☐ Форма варианта отчета
- ☐ Форма настроек динамического списка
- ☐ Форма поиска

Имя:

Синоним:

Комментарий:

Расширенное представление:

Пояснение:

☒ Использовать стандартные команды

☒ Командная панель формы сверху

☐ Командная панель формы снизу

< Назад Далее > Готово Отмена Справка

После нажатия на кнопку «Готово» откроется окно настройки формы.

В правой верхней части окна определяется список реквизитов формы (того, что входит в понятие «Данные формы»).

В левой верхней части находится дерево элементов. Если нужно изменить какие-либо свойства элементов, определить обработчики событий, то делать это нужно, начиная с данного дерева. Корень данного дерева определяет саму форму.

В нижней части находится область предварительного просмотра (какой внешний вид имеет форма в результате всех действий).

На закладке «Модуль» (ярлыки находятся в самом низу окна) можно прописывать код на встроенном языке системы.

Конфигурация: Параметры Учета

Форма

- Командная панель
 - НаименованиеОрганизации
 - ЮридическийАдрес
 - ФактическийАдрес
 - ИНН
 - КПП
 - СистемаНалогообложенияОрганизации
 - Руководитель
- Элементы
- Командный интерфейс

Реквизит

Использовать всегда	Тип
<input checked="" type="checkbox"/>	НаборКонстант (КонстантыНабор)

Реквизиты Команды Параметры

Записать и закрыть Записать Еще

Наименование организации:

Юридический адрес:

Фактический адрес:

ИНН:

КПП:

Система налогообложения организации:

Руководитель:

Учет по сериям: ☐

Учет по сканам: ☐

Форма Модуль

Используя одноименное свойство константы, каждой константе можно назначить свою основную форму.

Свойства: УчетПоСериям

Поиск (Ctrl+Alt+I)

▼ Основные:

Имя

Синоним

Комментарий

Тип ...

Модуль менеджера знач

▼ Данные:

Режим управления блоки

▼ Представление:

Использовать стандарт ☒

Основная форма ...

Если свойство не использовать, но на каждую константу система будет создавать форму автоматически.

Так как форма является некоторым промежуточным объектом, в ней можно выделить две составляющие: реквизиты и элементы управления.

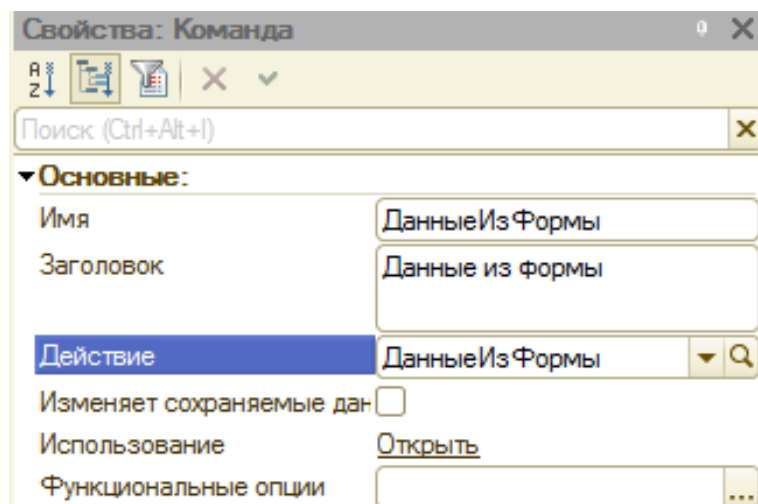
Реквизиты «ближе» к базе данных, элементы управления «смотрят» на пользователя.

Когда пользователь хочет что-либо «посмотреть», он дает команду на открытие этой формы. Система создает экземпляр формы и производит чтение значений объектов. Прочитанные значения «копируются» в реквизит формы объекта. Форма отправляется клиентскому приложению. Элементы управления отображают значения объектов. Связь элемента управления со значением свойств реквизита осуществляется через свойство «Данные» элемента.

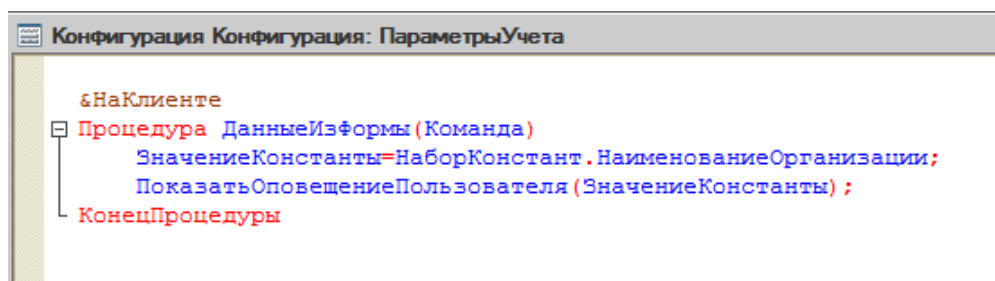
При настройке логики работы формы принят следующий подход: элементы управления выступают источниками событий. Программист описывает процедуры – обработчики событий. В обработчиках событий он работает с реквизитами формы. Измененные в обработчиках значения реквизитов автоматически отображаются элементами управления.

Получается, что в контексте формы есть «собственная копия» данных из базы.

На закладке «Команды формы» окна настройки формы создадим команду «ДанныеИзФормы».



Определим обработчик события следующим образом:



Подобным образом определим команду «ДанныеИзБазы».

```

    &НаСервереБезКонтекста
    □ функция ДанныеИзБазыСервер ()
        ЗначениеКонстанты=Константы.НаименованиеОрганизации.Получить ();
        Возврат (ЗначениеКонстанты) ;
    - Конецфункции

    &НаКлиенте
    □ Процедура ДанныеИзБазы(Команда)
        ЗначениеКонстанты=ДанныеИзБазыСервер ();
        ПоказатьОповещениеПользователя (ЗначениеКонстанты) ;
    - КонецПроцедуры

```

Каждая процедура, функция или объявление переменной модуля формы должны предваряться одной из следующих директив:

1. &НаКлиенте – процедура/функция выполняется на стороне клиента, а переменная существует все время жизни клиентской части управляемой формы. Из клиентского метода допустимыми являются вызовы клиентских, серверных и серверных внеконтекстных методов.

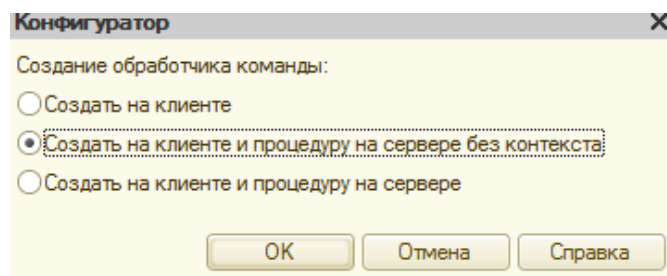
2. &НаСервере – процедура/функция выполняется на стороне сервера, а переменная существует только во время вызова выполнения серверного или серверного внеконтекстного вызова. Для серверных методов допустимыми являются вызовы серверных и серверных внеконтекстных методов.

3. &НаСервереБезКонтекста – процедура/функция исполняется на сервере вне контекста формы. Переменные не могут быть внеконтекстными. В таких методах недоступен контекст формы (включая данные формы). Допустимыми являются вызовы только других внеконтекстных методов. При вызове этих методов не выполняется передача данных формы на сервер и обратно. Применение внеконтекстных методов позволяет существенно уменьшить объем передаваемых данных при вызове серверной процедуры из среды клиентского приложения.

4. &НаКлиентеНаСервереБезКонтекста – процедура/функция может исполняться в управляемом клиенте или на сервере, при этом контекст формы не доступен.

Отсутствие директивы компиляции перед процедурой означает использование директивы «&НаСервере».

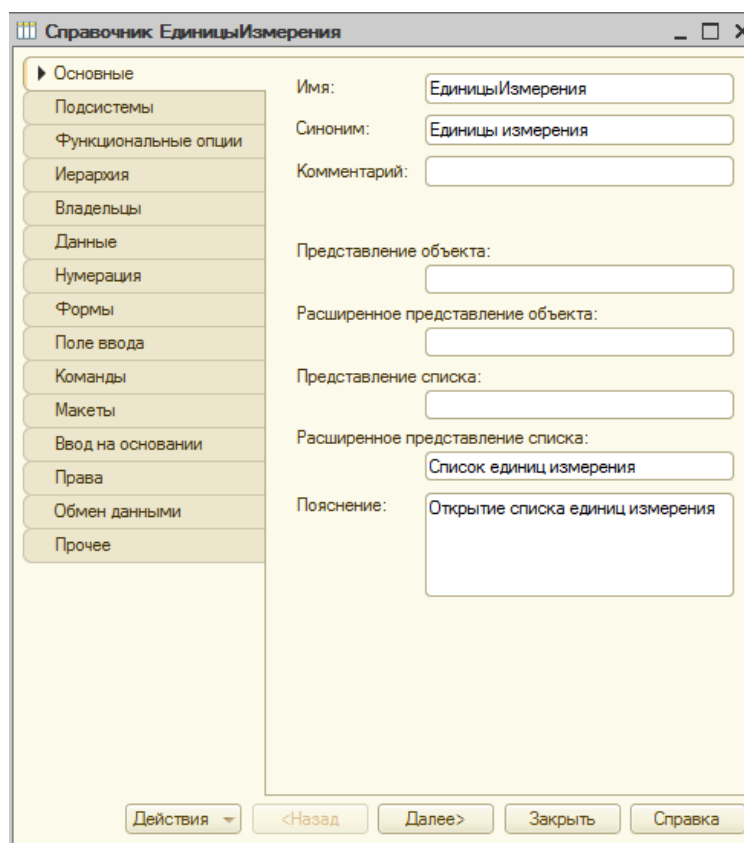
При создании обработчиков событий можно сразу выбирать: создается одна клиентская процедура, или еще создается процедура с местом исполнения – сервер.



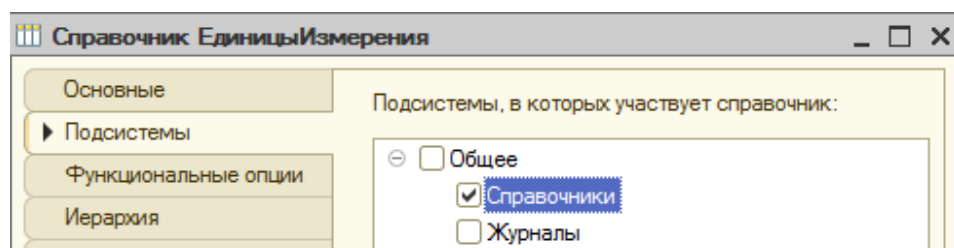
Справочники

Процесс создания и работы со справочниками подробно описан в предыдущей лабораторной работе, познакомимся подробнее с процессом работы с формами.

Создайте самостоятельно справочник «Единицы измерения», редактирование в списке.



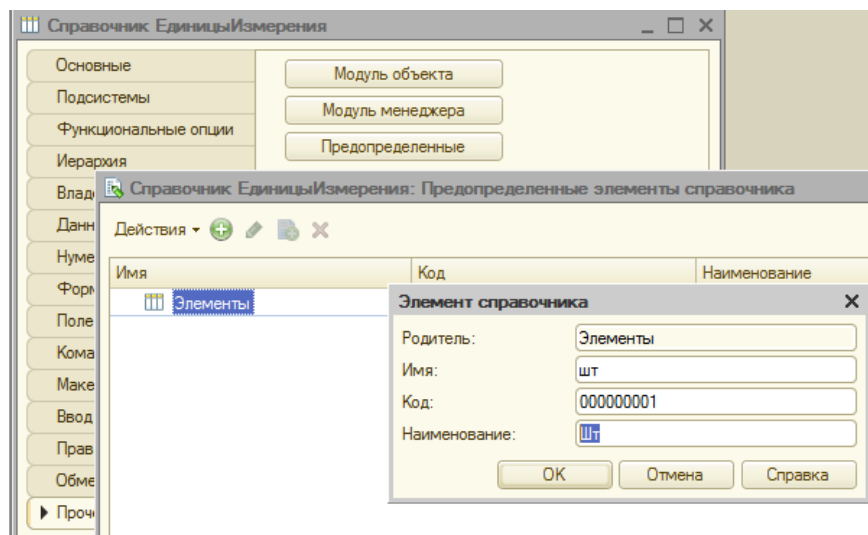
Один из важных этапов: отнесение объектов конфигурации к какой/каким-либо подсистемам.



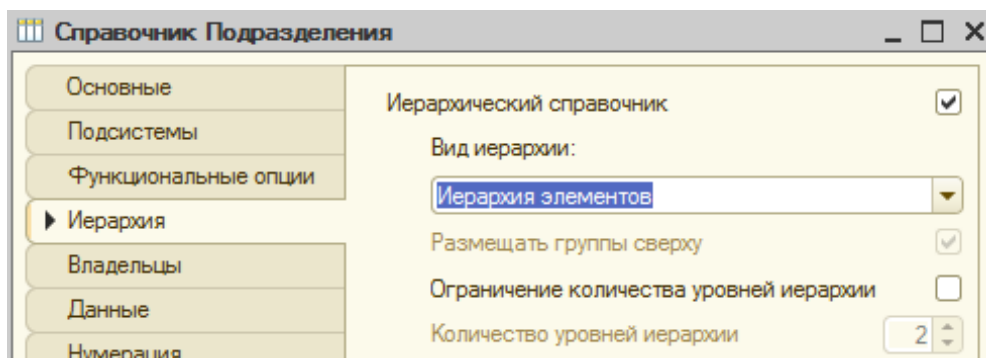
Отнесем справочник к подсистеме «Справочники», подчиненной системе «Общие».

На закладке «Данные» можно настроить длину кода и наименования, а также переопределить свойства стандартных реквизитов справочника.

На закладке «Прочие» определим предопределенные элементы. Добавляются они по одноименной кнопке, в форме, открываемой при нажатии на кнопку «Предопределенные».



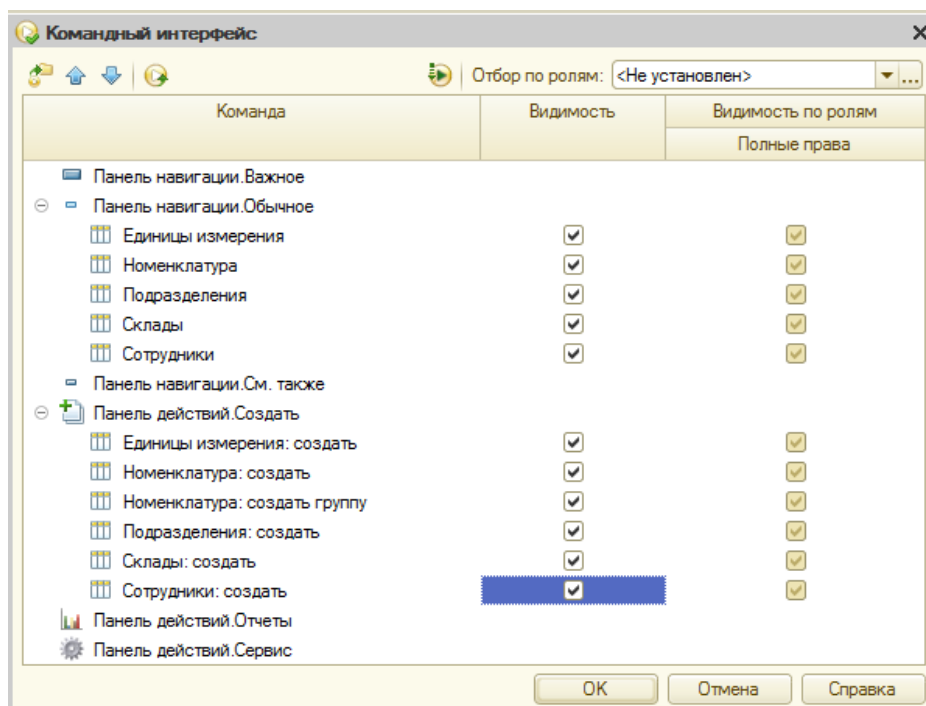
Создадим еще один справочник «Подразделения».



Данный справочник необходимо отнести к той же подсистеме, что и «Единицы измерения».

После установки флага «Иерархический справочник» в таблице справочника появляется поле «Родитель». В этом поле содержится ссылка на другую запись из этой же таблицы. Выбранный вид иерархии «Иерархия элементов» подразумевает то, что все записи в справочнике равнозначны, и любая запись может выступать «родителем» для другой записи.

Реализуем возможность создания элементов ранее определенных справочников не из командной панели списков, а напрямую с панели действий. Для этого нужно зайти в настройки командного интерфейса подсистемы «Справочники».

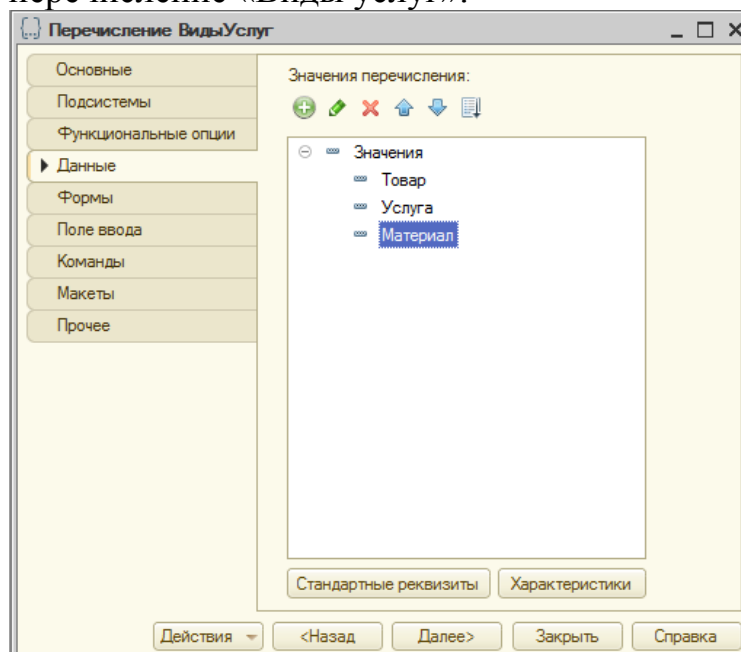


Перечисления

Иногда возникает необходимость задать в системе некий линейный и не изменяемый список. Это может быть список видов номенклатуры, операций того или иного документа, времен года, сторон света и т.п.

Для решения подобных задач предназначен такой объект, как перечисление. Перечисления создаются в соответствующей ветке дерева объектов конфигурации. После создания данного объекта конфигурации на закладке «Данные» определяется нужный перечень значений. В режиме исполнения это список не подлежит ни исправлению, ни переопределению состава. Пользователь может либо выбрать значение из этого списка, либо сбросить выбор.

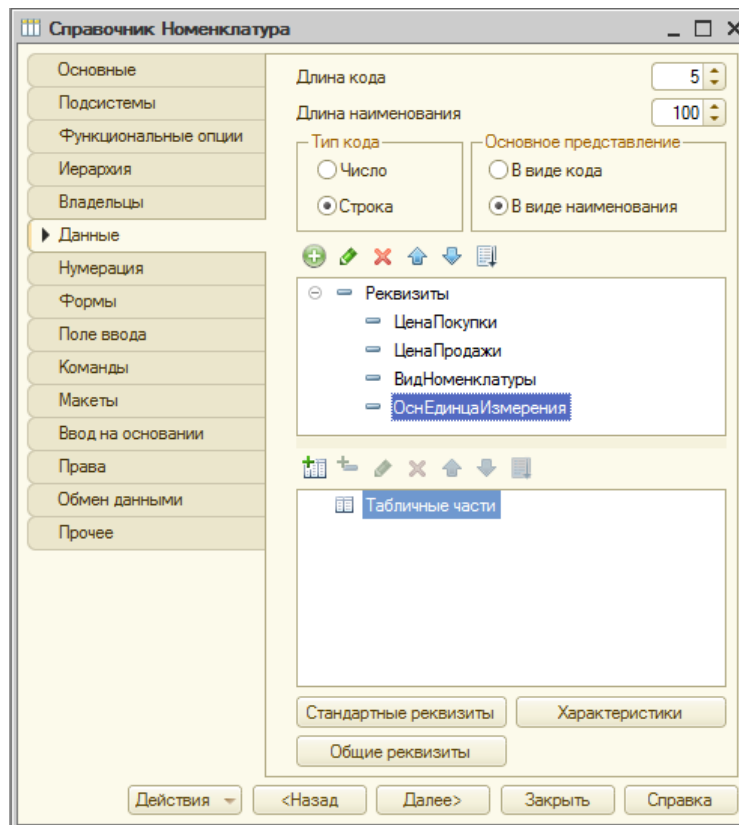
Создайте перечисление «Виды услуг».



Иерархия групп

При установке вида иерархии в значение «иерархия групп» в таблице справочника появляется дополнительно поле «ЭтоГруппа» (тип «Булево»). Все записи в справочнике начинают делиться на две категории: группы и элементы. Группы и элементы могут характеризоваться разными свойствами. При таком виде иерархии в качестве родителей могут выбираться только группы.

В справочнике «Номенклатура» на закладке «Данные» определите дополнительные реквизиты.

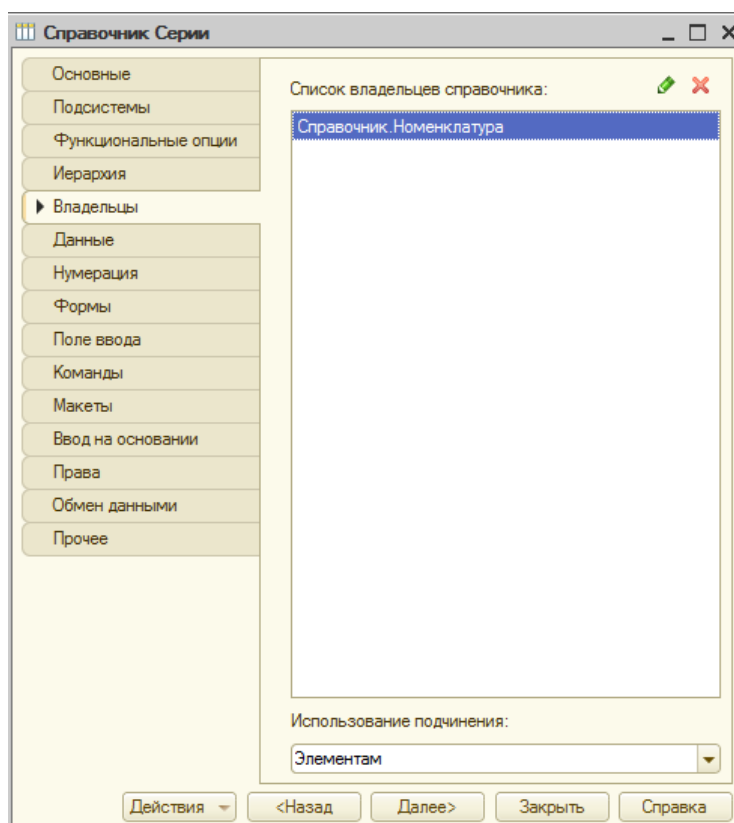


Создадим форму элемента справочника. В дереве элементов управления определим дополнительный элемент «Группа Обычная группа без отображения», для него определим настройки.

Элемент формы «ВидНоменклатуры» оформить как переключатель с видом переключателя «Тумблер». Потребуется работа со свойствами «Вид», «Вид переключателя» и «Список выбора».

Подчиненные справочники

Создадим справочник «Серии». На закладке «Владельцы» укажем, что справочник «Номенклатура» будет являться владельцем создаваемого справочника.



После подчинения в таблице справочника появится поле «Владелец». Это поле содержит ссылку на запись из другой таблицы. С точки зрения пользователя на справочник накладывается дополнительное ограничение: нельзя записать элемент подчиненного справочника с невыбранным владельцем. Поэтому на данном этапе заполнять подчиненный справочник проще, начиная с объекта – владельца.

Табличные части

Создадим справочник «Физические лица», без иерархии, без подчинения.

Также нужно создать табличную часть «Трудовая деятельность».

На закладке «Данные», воспользовавшись кнопкой «Стандартные реквизиты», переопределим синоним реквизита «Наименование» на «ФИО».

Создайте форму списка справочника.

После этого создадим форму элемента справочника и приведем ее к желаемому виду.

Создадим группу без отображения, разместим в ней код и наименование. Создадим обычную группу, разместим в ней элементы формы «Фамилия», «Имя», «Отчество».

Свойства: Группа

Поиск (Ctrl+Alt+I)

▼ Основные:

Имя: Группа5

Заголовок: ФИО полностью

Вид: Обычная группа

Видимость: ☒

Пользовательская видимость: Открыть

Доступность: ☒

Только Просмотр: ☐

Разрешить Изменение Со: ☒

Поведение: Свертываемая

Заголовок Свернутого От: ...

Свернута: ☐

Отображение Управлени: Гиперссылка заголовка

Отображение: Слабое выделение

Отображать Отступ Слева: ☒

Группировка: Вертикальная

Ширина Подчиненных Эле: Авто

Путь К Данным Заголовка: ... X

▼ Использование:

Сочетание Клавиш: X

Подсказка:

Отображение Подсказки: Авто

Имя объекта
Имя, Name

Также требуется, чтобы на форме было две странички. На первой располагались реквизиты справочника, на второй страничке – табличная часть и командная панель этой табличной части. Добавим в форму новый элемент, тип элемента – «Страницы».

Сделаем добавленную группу текущей и добавим два новых элемента управления с типом «Страница».

Останется «перетащить» элементы управления на нужные страницы, чтобы добиться внешнего вида следующей формы.

Общие Трудовая деятельность

Код: ФИО:

ФИО полностью

Фамилия:

Имя:

Отчество:

Дата рождения:

Печатные формы

Для формирования печатных форм в системе используются два объекта: **«Макет»** и **«Табличный документ»**.

Суть алгоритма печати:

Макет содержит набор именованных областей. В каждой именованной области определены элементы оформления и набор параметров, параметров расшифровки. В процедуре, реализующей печать, первоначально создается «ТабличныйДокумент». Из макета получается очередная именованная область. Выбранные данные записываются в параметры, параметры расшифровки области. Заполненная данными область включается в табличный документ. В конце алгоритма табличный документ открывается.

Для создания печатной формы воспользуемся соответствующим конструктором, но предварительно создадим группу команд «Печать» (внутри ветви «Общие» дерева объектов конфигурации).

Свойства: Печать

Поиск (Ctrl+Alt+I)

▼ Основные:

Имя

Синоним

Комментарий

Категория

▼ Представление:

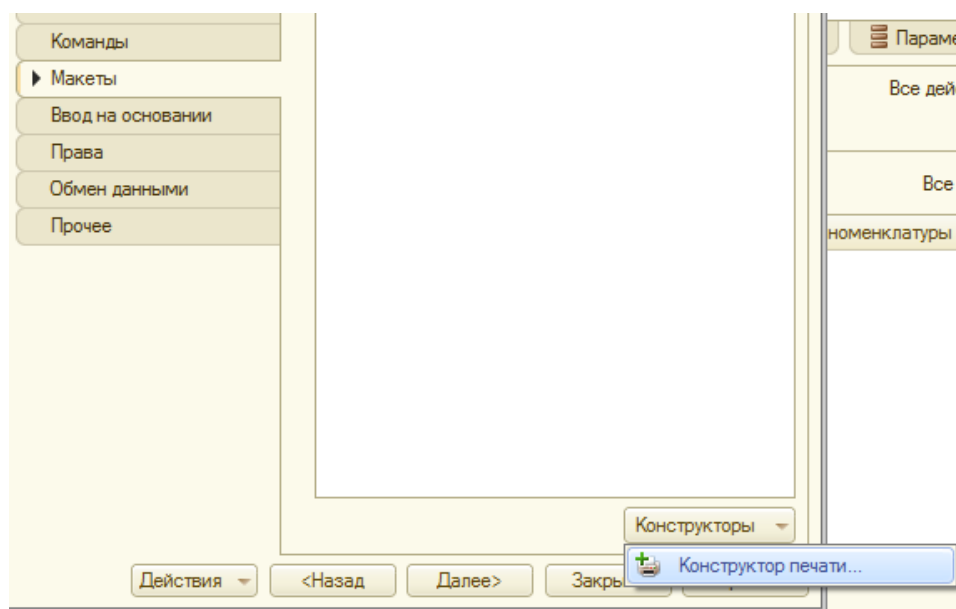
Отображение

Подсказка

Картинка

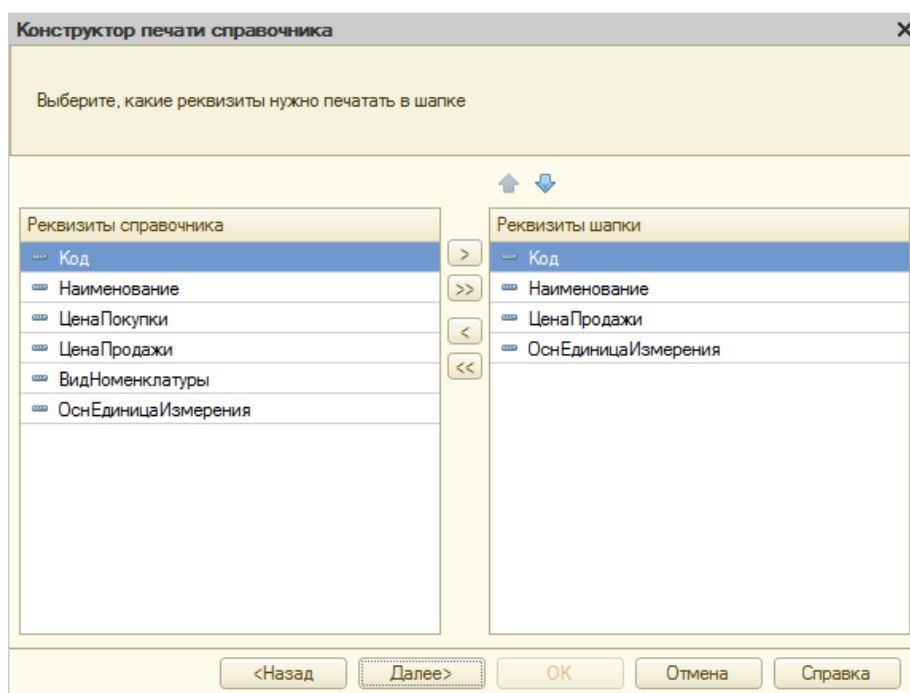
Один из важных моментов при создании группы это определение категорий.

В окне объекта конфигурации «Номенклатура» на закладке «Макеты» по кнопке «Конструктор печати» откроем одноименный конструктор.

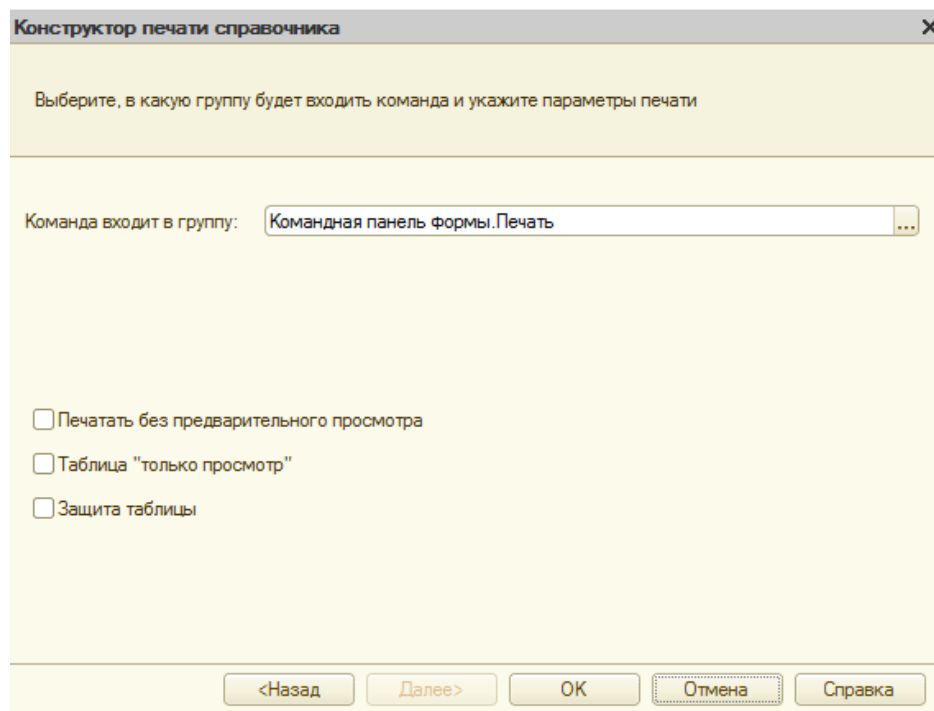


В открывшейся форме указываем, что создается новая команда. На этом же этапе можно переопределить имя создаваемой команды.

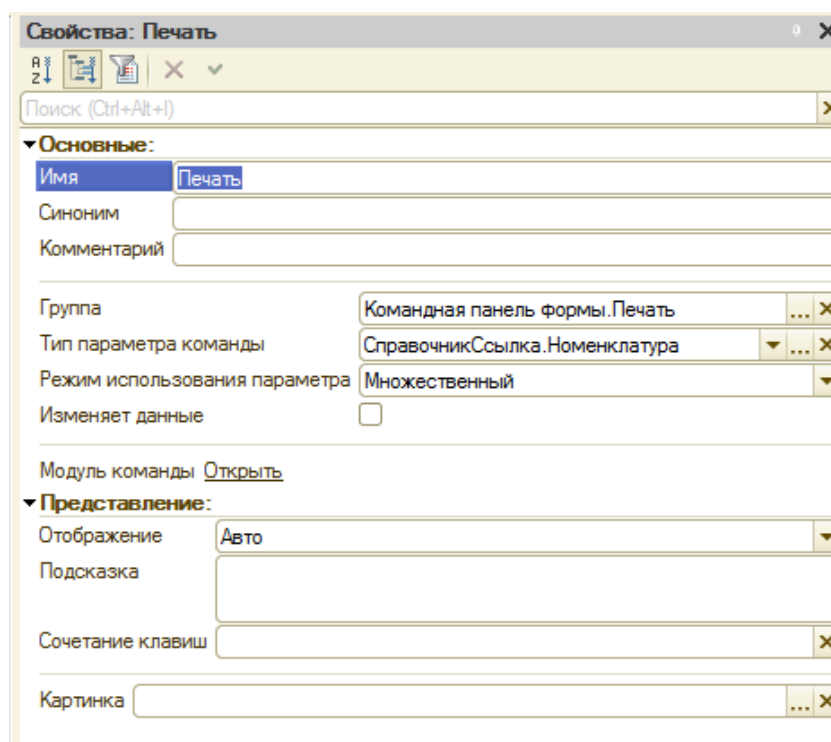
На следующем этапе из реквизитов справочника выбираем те, которые хотим видеть в печатной форме.



На последнем этапе указываем, что создаваемая конструктором команда будет входить в ранее созданную группу «Печать».



В результате работы конструктора у объекта конфигурации была создана команда.



В модуле команды определен следующий код.

```

&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)

 Справочники.Номенклатура.Печать(ТабДок, ПараметрКоманды);

КонецПроцедуры

В модуле менеджера справочника:

Процедура Печать(ТабДок, Ссылка) Экспорт

 //{{_КОНСТРУКТОР_ПЕЧАТИ(Печать)

 Макет = Справочники.Номенклатура.ПолучитьМакет("Печать");

 Запрос = Новый Запрос;

 Запрос.Текст =

 "ВЫБРАТЬ

 | Номенклатура.Код,

 | Номенклатура.Наименование,

 | Номенклатура.ОснЕдиницаИзмерения,

 | Номенклатура.ЦенаПродажи,

 | Номенклатура.ВидНоменклатуры

 |ИЗ

 | Справочник.Номенклатура КАК Номенклатура

 |ГДЕ

 | Номенклатура.Ссылка В(&Ссылка)";

 Запрос.Параметры.Вставить("Ссылка", Ссылка);

 Выборка = Запрос.Выполнить().Выбрать();

 ОбластьЗаголовков = Макет.ПолучитьОбласть("Заголовок");

 Шапка = Макет.ПолучитьОбласть("Шапка");

 ТабДок.Очистить();

 ВставлятьсяРазделительСтраниц = Ложь;

 Пока Выборка.Следующий() Цикл

 Если ВставлятьсяРазделительСтраниц Тогда

```

//ТабДок.ВывестиГоризонтальныйРазделительСтраниц();
    КонецЕсли;

    ТабДок.Вывести(ОбластьЗаголовков);

    Шапка.Параметры.Заполнить(Выборка);

    Шапка.Параметры.ВидНом      =      "Какой-то      вид
номенклатуры";

    ТабДок.Вывести(Шапка, Выборка.Уровень());

    ВставлятьРазделительСтраниц = Истина;
    КонецЦикла;
//}}
КонецПроцедуры

```

Проверьте механизм на практике.

Задание.

Создайте или отредактируйте (если они ранее были созданы) следующие справочники:

- 1. «Склады». Справочник без иерархии, без подчинения. Реквизитов и табличных частей нет.**
- 2. «Контрагенты». Справочник иерархический (иерархия групп и элементов), без подчинения. Дополнительный реквизит «НаименованиеПолное», тип «Строка» 300 символов.**
- 3. «КонтактныеЛица». Справочник без иерархии, подчинен справочнику «Контрагенты». Дополнительный реквизит «Телефон», тип «Строка» 15 символов.**
- 4. «Должности». Справочник без иерархии, без подчинения. Реквизитов и табличных частей нет. В нем определены 3 предопределенных элемента с именами «Бухгалтер», «ГлавныйБухгалтер», «Кассир».**

3. Контрольные вопросы.

- 1) Для чего нужен объект конфигурации «Константы»?
- 2) Для чего нужен объект конфигурации «Перечисление»?
- 3) Для чего нужен объект конфигурации «Справочник»?
- 4) Какие типы данных используются в системе 1С Предприятие?
- 5) Перечислите основные виды объектов конфигурации и их характеристики.
- 6) Что такое объекты конфигурации?