??????????
??????????

????? ???????????

Основа: "Разработка интерфейса прикладных решений на платформе 1С:Предприятие 8" от 2018 г., платформа 8.3

ВАЖНО. Отказ от модальности - необходимое условие для работы в веб-клиенте и на мобильных устройствах.

Разработчик создает описание блоков интерфейса и события, за формирование вида отвечает платформа. Описание включает факт наличия элемента, свойства элемента и принадлежность к блоку, точное попиксельное размещение недоступно. Факт отображения зависит от прав пользователя.

Форма как программный объект существует на клиенте и на сервере. При реализации логики в общем случае происходит передача контекста с клиента на сервер и обратно.

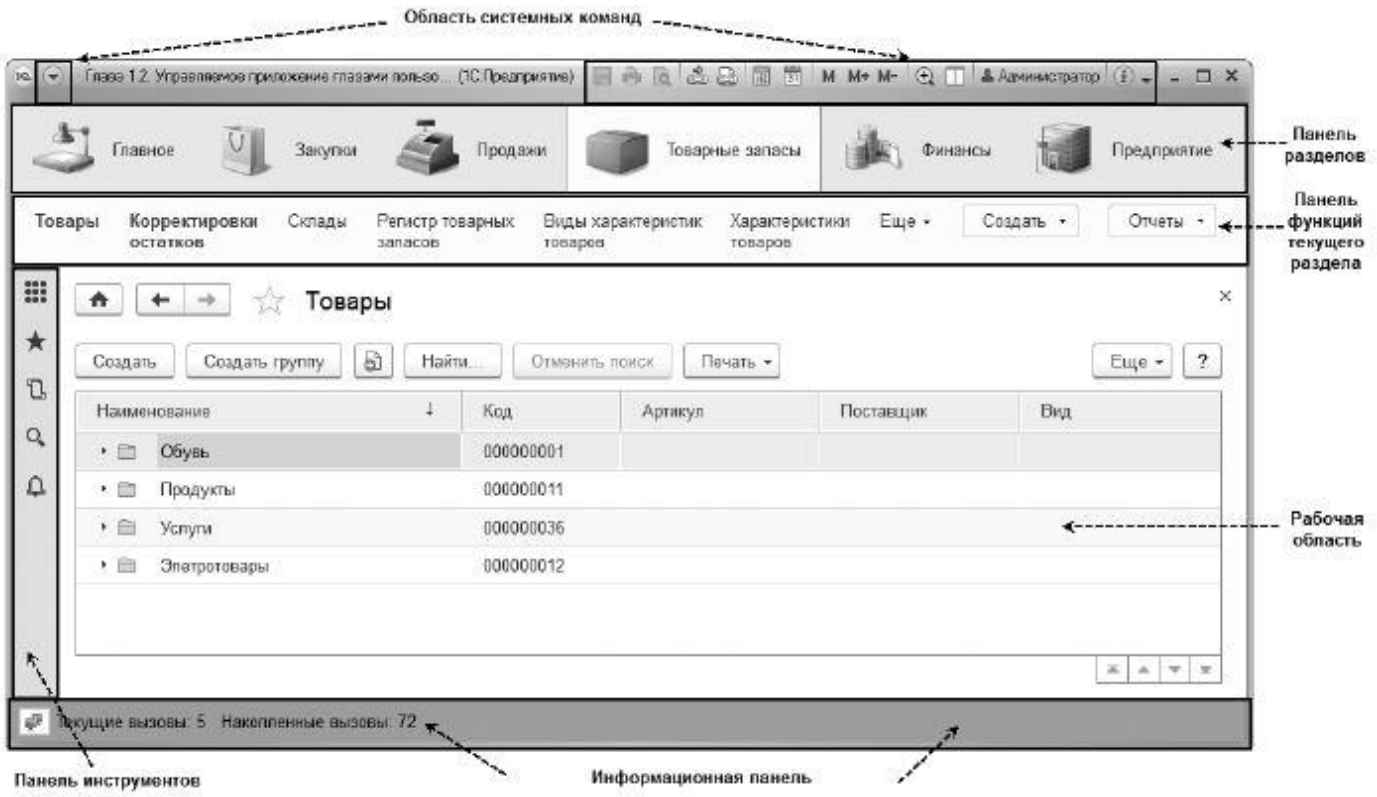
Существуют реквизиты формы и параметры формы. Реквизиты формы связаны с отображаемыми элементами интерфейса и существуют все время существования формы. Параметры нужны для управления функциональностью формы и существуют в момент создания и открытия формы. Затем все параметры, кроме ключевых, удаляются.

Есть 4 режима основного окна: Обычный, Рабочее место, Полноэкранное рабочее место, Киоск. Не нашел и не попробовал, статья Далее обычный режим.

Основной тип интерфейса - Такси. Блочный однооконный, в web версии по виду и функциям аналогично desktop версии. Пользователь может донастраивать сам почти весь интерфейс. Блоки интерфейса:

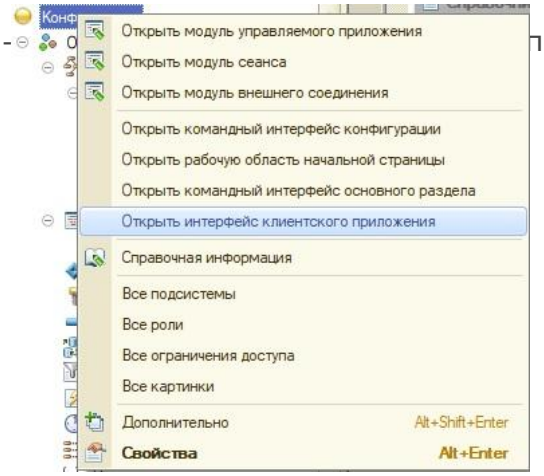
Блок	Информация
Область системных команд	Общие действия по управлению. Не зависит от конфигурации. Не настраивается пользователем и разработчиком.

Блок	Информация
Панель разделов	Главное + подсистемы первого уровня. Не настраивается пользователем. **Настройка Главной:** ПКМ на корне - Рабочая область начальной страницы. Не зависит **Настройка набора видимых подсистем:** ПКМ на корне - Открыть командный интерфейс конфигурации. Делится по ролям пользователей. Т е можно в правах ролей задать доступ к подсистеме в целом, а можно скрыть из списка. Роли: жесткое ограничение. Сокрытие означает, что оно будет не видно, но часть функций можно включить в другие системы/подсистемы и будет видно.
Панель функций текущего раздела	4 части: Элементы и 3 выпадающих списка: Еще, Создать, Отчеты, Сервис. **Элементы:** Сначала списки. Важное выделены жирным, самые первые. Все, что не вошло, попадает в выпадающий список Еще. Зависит от размера окна, чем уже - больше элементов в Еще. Элементы из подчиненных подсистем независимо суммируются с набором текущей, могут повторяться, деление на подчиненные не видно. Увидеть деление на подсистемы через блок Панель инструментов.
Панель инструментов	Состоит из Меню функций, Избранное, История, Поиск, Центр оповещений. В меню функций видно деление на подчиненные подсистемы. Через шестеренку пользователь может настроить под себя. Если оповещений в принципе нет - элемент Центр оповещений отсутствует.
Панель избранного	По умолчанию выключена. Для включенной панели нельзя настроить отображение.
Панель истории	По умолчанию выключена. Для включенной панели нельзя настроить отображение.
Панель открытых	По умолчанию выключена. Одна строка внизу.
Рабочая область	Отображение элемента из выбранного блока.
Информационная панель	Отображение режима работы и отладочной информации.



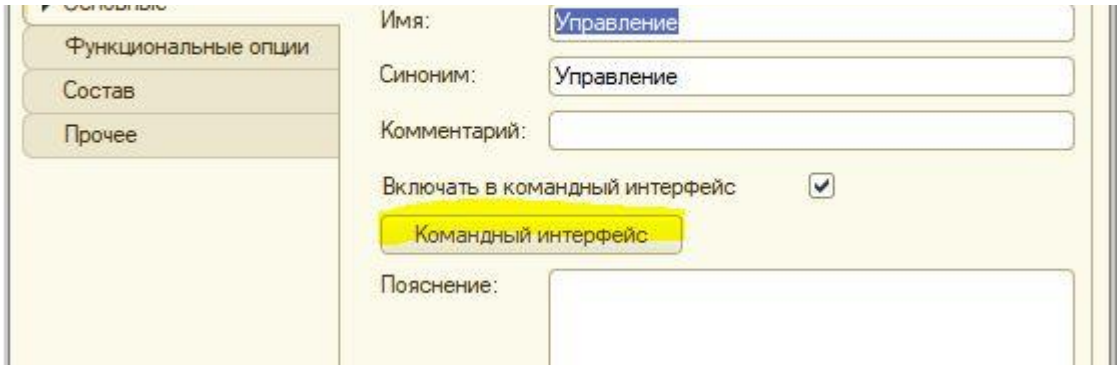
Настройка блочного состава пользовательского интерфейса.

ПКМ на корне конфигурации - приложения.



Настройка интерфейса подсистем

Настраивается в основном разделе настройки подсистемы, кнопка командный интерфейс.


Функциональные опции
Состав
Прочее
Имя:
Управление
Синоним:
Управление
Комментарий:
Включать в командный интерфейс
Командный интерфейс
Пояснение:

????? ???????????

Здесь и далее под формой понимается управляемая форма (часть кода на сервере, часть на клиенте).

Виды форм интерфейса и взаимосвязь параметров и Основного параметра реквизитов

Существуют формы объекта и произвольные формы.

Форма объекта - форма с заранее установленным реквизитом в соответствии с типом объекта (например в форме элемента это называется Объект и содержит реквизиты соответствующего элемента, в форме списка это Список).

В случае наличия возвращаемого значения (в форме выбора), возврат создается автоматически.

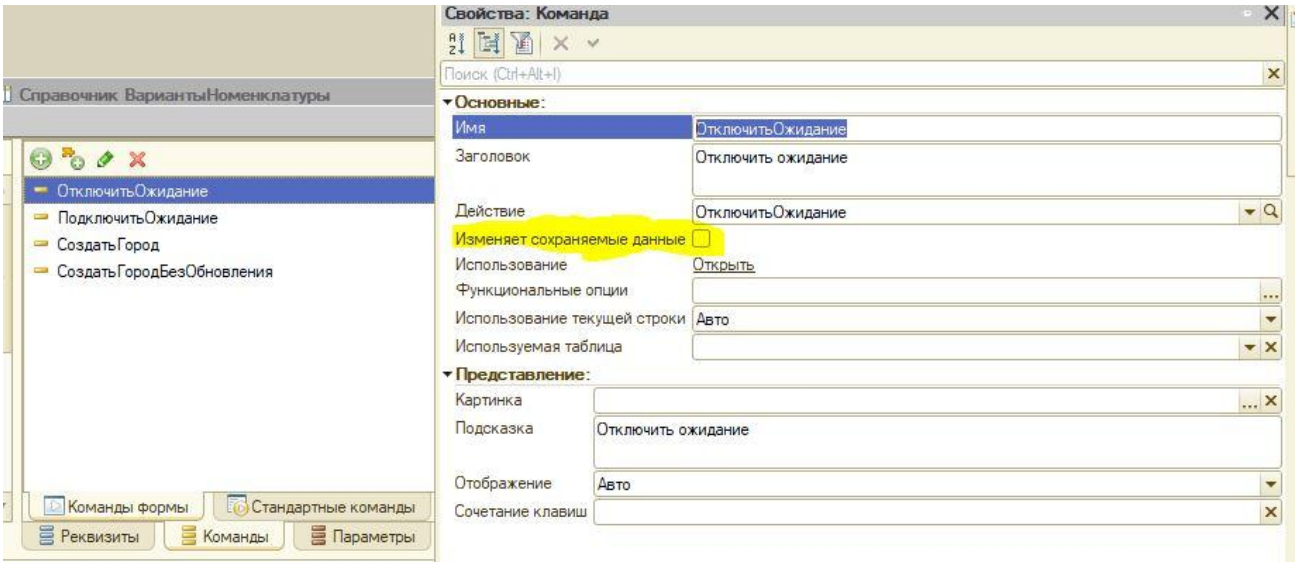
Поддерживается полная программная настройка жизненного цикла формы (Обработка входных параметров - Начальное заполнение - Обработка событий - Возврат выходных параметров).

Термин "заранее установленный реквизит" соответствует реквизиту формы с установленной опцией "Основной параметр".

Для структуры Параметры при программном открытии формы, ключ "Ключ" ссылается на основной параметр. Внутри формы он может называться как угодно, связь происходит по свойству "Основной параметр". В произвольной форме признак основной параметр можно установить, если в типе реквизита указать ...Объект, например СправочникОбъект.

Статья о формах

Есть критичная особенность обработки данных, связанная с механизмом блокировок и основным параметром. При установке в свойствах команды флага Изменяет сохраненные данные,



при попытке выполнения команды будут выполняться следующие действия:

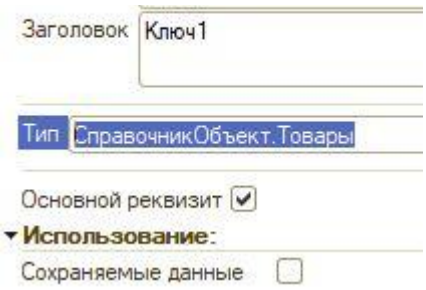
- Попытка заблокировать основной параметр. При неудаче команда не будет выполнена.
- Будет установлен признак изменения у формы (флаг «Модифицированность»).
- Для нового и несохранённого объекта будет произведена попытка записи. При этом пользователю будет задан вопрос о необходимости записи. Если ответ будет отрицательный — команда не будет выполнена.

Отсутствие этого флага не влияет на возможности модификации, но забота о целостности и непротиворечивости данных ложится на плечи разработчика.

Эксперимент. Бессмысленный с практической стороны, только для демонстрации взаимосвязи ключа "Ключ" и основного параметра формы. Справочник Города. В форму элемента добавлена кнопка и реквизит ссылка на товары. Обработка формы:

```
&НаКлиенте
Процедура КомандаОткрытияТоваров(Команда)
	Парам = Новый Структура;
	Парам.Вставить("Ключ", НужныйТоварСсылка);
	ОткрытьФорму("Справочник.Города.Форма.ФормаПростоТак", Парам);
КонецПроцедуры
```

В ФормеПростоТак такая настройка:

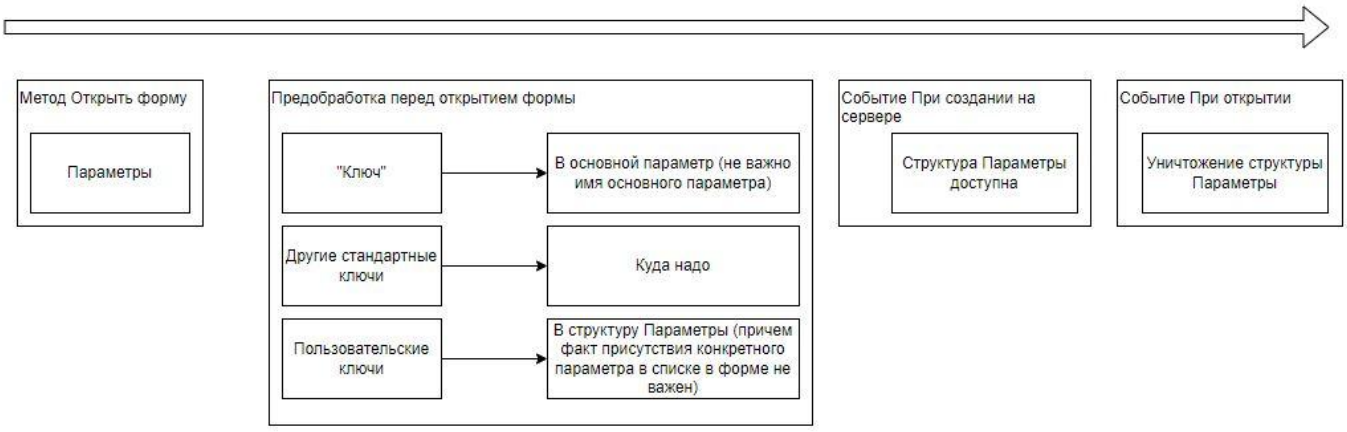


Имена параметров не совпадают, но когда он основной реквизит - выбранный товар открывается. Похоже, в этом случае ссылка автоматически преобразуется в объект. Если не основной - не открывается. А далее было получено парадоксальное (на момент написания этой строки) поведение: если реквизит основной, но в обработке формы меняю имя параметра на совпадающее название (Ключ1) - данные не подгружаются. После раскуривания умной книжки опять пришлось выдохнуть, улыбнуться, понять и простить разработчиков за использование одного и того же слова в разных контекстах и скрытое преобразование данных:

Взаимосвязь между структурой Параметры в методе Открыть форму (см. раздел Открытие другой формы), реквизитами и параметрами формы. Перед созданием формы, структура Параметры в методе Открыть форму сопоставляется с параметрами в разделе Параметры в форме и стандартными ключами.

При обработке события "При создании на сервере" доступна структура Параметры, в которой будут находиться все переданные параметры. Факт наличия конкретного параметра в списке параметров формы не важен (понять и простить). Указание в списке параметров влияет только на отображение этого параметра в списке контекстной подсказки, но при указании несуществующего параметра ошибки не происходит, и если этот параметр был передан - он считается.

Параметр "Ключ" передается в реквизит со свойством "Основной параметр" напрямую и это свойство может быть выставлено только у одного реквизита. Если в обработке события "При создании на сервере" не сделать заполнение реквизитов (или каких-либо других действий), все пользовательские параметры будут проигнорированы и затем структура Параметры очищается.




```
&НаСервере
Процедура СообщитьИнфу()
	Сообщить(Параметры.Реквизит3);
КонецПроцедуры

&НаКлиенте
Процедура Команда1(Команда)
	СообщитьИнфу();
КонецПроцедуры

&НаСервере
Процедура ПриСозданииНаСервере(Отказ, СтандартнаяОбработка)
	Реквизит1 = Параметры.Реквизит3;
КонецПроцедуры
```

Команда1 привязана к кнопке на форме. В параметрах формы - Реквизит2. Реквизит3 в параметрах формы отсутствует. В реквизитах формы - Реквизит1. Следующий код корректно открывает вышеуказанную форму и заполняет Реквизит1 значением 5:

```
&НаКлиенте
Процедура КомандаОткрытияФирм(Команда)
	Парам = Новый Структура;
	Парам.Вставить("Реквизит3", 5);
	ОткрытьФорму("Справочник.Города.Форма.ФормаПростоТак", Парам);
КонецПроцедуры
```

При выполнении процедуры СообщитьИнфу выдается ошибка отсутствия поля Реквизит3. То есть структура Параметры после создания формы остается, но пустая. Если перенести Сообщить(Параметры.Реквизит3); в Команда1, то ошибка такая же.

Передача реквизита формы в серверную процедуру

[Описание процедур]

Необходимость в данной технологии возникает, когда есть данные на форме, и нужно провести с ними манипуляции. Т е есть еще не записанные в базу данные и есть модули (общие, ...) и нужно передать туда данные без записи, модифицировать и отдать обратно в форму.

РеквизитФормыВЗначение() компилируется &НаСервере, контекст формы необходим. На клиенте не работает, поскольку в результате получаем прикладной объект.

ЗначениеВРеквизитФормы() для возврата данных на форму. Вторым параметром является тип значения, необязательный. Если обрабатываемый реквизит не является составным типом, то тип будет автоматически получен из реквизита формы. Иначе генерируется исключение времени выполнения.

```
&НаСервере
Процедура ЗаполнитьТЧПоДаннымПоследнегоДокумента(Контрагент)
            ДокОбъект = РеквизитФормыВЗначение("Объект");
            ДокОбъект.ЗаполнитьТЧПоДаннымПоследнегоДокумента(Контрагент);
            ЗначениеВРеквизитФормы(ДокОбъект, "Объект");
КонецПроцедуры
```

Точки хранения форм

Формы могут храниться в Общие - Общие формы и в объектах конфигурации, раздел Формы. Произвольную форму можно создать во всех точках хранения.

В общих формах можно создать формы констант, отчетов, динамических списков, поиска, истории данных.

В объектах конфигурации - набор форм в соответствии с типом объекта. Создание произвольной формы в объекте конфигурации технически возможно, актуально если логически данная форма выполняет действия, связанные с этим объектом. Должны быть аргументы для размещения произвольной формы в объекте, в общем случае логичнее произвольную форму разместить в общих формах. Плюс размещения произвольной формы в

объекте - возможность выбора этой формы в качестве формы чего-либо. Однако, стандартная форма ничем не хуже произвольной, кроме уже настроенного объекта и привязанного к нему свойству Основной реквизит.

Модальные и обычные окна

Модальные окна в 1С - это такой тип окна, вызываемый в среде 1С, который блокирует остальной интерфейс 1С 8. Сейчас устаревшая технология.

Способы взаимодействия между формы

Посмотреть интерактивное взаимодействие и оповещения

1. ОткрытьМодально, параметр закрытия, метод Закрыть. - устаревший способ, не использовать.

```
//В первой форме
ВозвращаемоеЗначение = ДругаяФорма.ОткрытьМодально().
//В другой форме
ЭтаФорма.Закрыть(ВозвращаемоеЗначениеВПервуюФорму)
```

2. Реквизиты формы (или свойства расширения формы). Реквизиты формы видимы снаружи как свойства объекта «Форма».

```
//В первой форме
ДругаяФорма.ИмяРеквизита = Значение.
```

3. Через ЭлементыФормы.

```
//В первой форме
ДругаяФорма = ДокументСсылка.ПолучитьФорму().
ДругаяФорма.ЭлементыФормы.ИмяПоля.Значение = Значение.
```

4. Оповещения.

```
//В первой форме оповещаются все открытые формы
Оповестить("ИмяСобытия", "Форма1", Данные);

//В другой форме следует назначить обработчик события оповещения.
Процедура ОбработкаОповещения(Событие, Источник, Данные)
   Если Событие = "ИмяСобытия" И Источник = "Форма1" Тогда
   КонецЕсли;
КонецПроцедуры
```

5. Экспортные переменные модулей форм. Они становятся видимы снаружи примерно как реквизиты форм.

```
//В первой форме
ДругаяФорма.ИмяПеременной = Значение;
//В другой форме
Перем ИмяПеременной Экспорт;
```

6. Параметры сеанса.

```
//В первой форме
ПараметрыСеанса.ИмяПараметра = Значение
//В другой форме
Если ПараметрыСеанса.ИмяПараметра = Значение Тогда КонецЕсли
```

Работа с элементами формы

Доступ к выбранному элементу табличной части. Контактные лица - имя табличной части.

```
&НаКлиенте
Процедура СообщитьВыбраннуюСтроку(Команда)
	ВыбранныйКонтакт = Элементы.КонтактныеЛица.ТекущиеДанные;
	Если ВыбранныйКонтакт = Неопределено Тогда
		Сообщить("Контактов не существует");
	Иначе
		Сообщить(ВыбранныйКонтакт.Телефон);
	КонецЕсли;
КонецПроцедуры
```

Проверка заполнения поля. Важный момент, потратил минут 30. ! Код нужно тестировать !

```
ЗначениеЗаполнено(Поле)
```

Открытие формы.

Статья по открытию форм, но стоит проверить данные в статье.

Существуют объекты ссылочного типа (Справочники, элементы справочника и т д) и нессылочного, например элементы регистра. Естественно, если в форме присутствует ссылка на объект, их можно передать в создаваемую форму в качестве параметра.

Способы открыть форму:

ОткрытьФорму	ПолучитьФорму	ПоказатьЗначение
Основной универсальный способ. Сразу открывает форму.	Получаем форму без открытия, затем метод Открыть. Как ОткрытьФорму.	Типа упрощение, негибкое

Метод ОткрытьФорму. Универсальный способ. Полный список параметров метода «ОткрытьФорму»: ИмяФормы, Параметры, Владелец, Уникальность, Окно

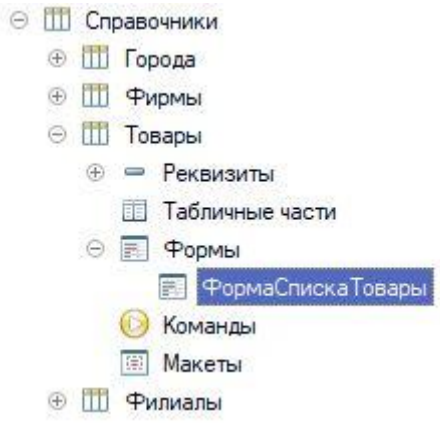
Свойство ИмяФормы. Пример открытия формы с передачей только имени формы. Параметром передается строка, подсказки нет, поэтому внимательно к именам (Справочник в единственном числе, Форма в единственном числе).

```
ОткрытьФорму("Справочник.Товары.ФормаЭлемента");//Основная форма
ОткрытьФорму("Справочник.Товары.Форма.ПроизвольнаяФормаТовара");// + слово Форма для
неосновной формы
```

Нюансы: формы константы нет, нужно сначала создать общую форму констант.

Пример. Кнопка, к которой привязано действие, размещена в форме элемента справочника Города.

```
&НаКлиенте
Процедура КомандаОткрытияТоваров(Команда)
ОткрытьФорму("Справочник.Товары.Форма.ФормаСпискаТовары");
КонецПроцедуры
```



Свойство «Параметры» Тип Структура. Передать туда можно все, что можно передать с клиента на сервер.

Если объект ссылочный (например конкретный документ, на который есть ссылка), то необходимо в поле Ключ передать ссылку.

Пример: есть справочник, в котором есть табличная часть КонтактныеЛица. У табличной части есть реквизит ДокументТестовый типа ДокументСсылкаПоступлениеТоваров. Процедура обработки нажатия на кнопку:

```
&НаКлиенте
Процедура ОткрытьДокумент(Команда)
	ВыбранныйКонтакт = Элементы.КонтактныеЛица.ТекущиеДанные;
	Если ВыбранныйКонтакт = Неопределено Тогда
		Сообщить("Контактов не существует");
	Иначе
		Док = ВыбранныйКонтакт.ДокументТестовый;
		Если ЗначениеЗаполнено(Док) Тогда
			Парам = Новый Структура;
			Парам.Вставить("Ключ", Док);
			ОткрытьФорму("Документ.ПоступлениеТоваров.ФормаОбъекта", Парам);
		Иначе
			Сообщить("Документ не привязан.");
		КонецЕсли
	КонецЕсли;
КонецПроцедуры
```

При передаче пустого ключа создастся новый документ.

В случае, если нужно создать новый объект на основании существующего, то нужно добавить параметр "Основание" со ссылкой на объект, на основании которого нужно сделать новый документ. В этом случае вызывается процедура «ОбработкаЗаполнения» модуля объекта. Эта процедура имеет единственный параметр «Основание». Если мы в параметры вставим ключ «Основание», то он будет передан в процедуру «ОбработкаЗаполнения».

Если нужно открыть форму создания группы, то в параметрах указать "ЭтоГруппа" Истина.

События формы

Интересная статья

События при создании формы (сверху вниз)

Модуль формы клиент	Модуль формы сервер	Модуль объекта сервер
	При чтении на сервере (существующий объект)	Обработка заполнения (новый объект)
	При создании на сервере	
При открытии		

Прикладные задачи

Передача параметров в произвольный запрос динамического списка. При открытии формы передаются конкретные значения параметров, в обработчике события формы При создании на сервере эти значения устанавливаются параметрами запроса в динамическом списке.

?????????? ??????????

Цвет

```
ЭлементыФормы.ПолеВвода1.ЦветФонаПоля = WebЦвета.Красный;
ЭлементыФормы.ПолеВвода1.ЦветФонаПоля = Новый Цвет(51,51,51);
```

Цвет по-умолчанию задается пустым Цвет():

```
Если Объект.Завершен Тогда
  Элементы.ПроцессЗавершенНадпись.ЦветТекста = Новый Цвет();
КонецЕсли;
```

Общая информация

Программное создание элементов оформления

Но создавать программно команды нельзя, а можно создавать ссылки на созданные команды.

Элементы оформления и представления реквизитов

Под элементом оформления будем понимать элемент, создаваемый без участия реквизита, а под представлением реквизита - элемент оформления, создаваемый из реквизита. Это почти одно и то же, но есть нюансы: есть 3 набора элементов, которые нельзя создать из реквизитов: Группа, Декорация и Дополнение элемента формы. А представления реквизитов - это настроенный под отображение реквизита определенного типа, один из трех оставшихся элементов оформления (поле, кнопка, Таблица). Так что привязанные к реквизиту элементы оформления проще создавать из реквизитов путем перетаскивания на форму. Дальше слово свойство подразумевает свойство элемента оформления.

Группы-Обычная группа контейнер компонентов, расположенных в соответствии с настройками (горизонтально или вертикально). Могут быть окружены рамкой, отображаться на общем фоне заданного цвета и иметь общий заголовок. Комбинируя свойства Поведение, горизонтально / вертикально и дочерние группы можно сделать очень навороченный интерфейс.
Группы-Страницы. Для страничного отображения. Внутри нужно добавить группы для отображения в виде страницы. Тип у дочерней группы становится Страница)
Группы-Командная панель. Отображает команды, созданные в форме или у существующих элементов (у кого команды предусмотрены). Хотя у меня не отобразилась команда в списке возможных, так что хз.

Декорация-Надпись и Декорация-Картинка есть события нажатие и обработка навигационной ссылки.

Дополнение элемента формы. Можно указывать строку поиска, управление поиском. Нужно указывать табличный элемент, по которому будет происходить поиск. Если нужно вынести блок поиска в другое место от шапки таблицы.

Общие свойства элементов оформления

Заголовок. Работает для простых типов - подпись около элемента. Для сложных - не работает.
Важность при отображении. Работает в случае мобильного клиента. Группирует менее важные элементы в выпадающий список.

Расположение. Ширина и Высота 0 - авто, однако старается вместить все элементы, комбинация например горизонтальная всегда и ширина 10 для 4 кнопок не работает. Однако например вертикальная и 10 сжимает каждую из 4 кнопок.

Детали для элемента Группа

Вид. Соответствует типам групп Обычная группа, Страницы, Командная панель.
Поведение. Два дополнительных способа - Свертываемая и Всплывающая.
Свертываемая - позволяет свернуть элементы. Должно быть задано свойство Заголовок (будет отображаться в раскрытом состоянии). Может быть задано свойство ЗаголовокСвернутогоОтображения и свойство Свернута. Свойство ОтображениеУправления (картинка или гиперссылка) определяет вид оформления заголовка.
Всплывающая - по умолчанию в виде ссылки, по клику всплывает дополнительное окно (не модальное).
Отображение. Степень акцента на группе при помощи размера шрифта заголовка и расстояния между группами, Нет соответствует группе без отображения.
Группировка. Порядок группировки вложенных элементов внутри группы.
Объединенная. Не понял, зачем нужно. Ведь если группа создается, то подразумевается, что элементы должны подчиняться правилам группы.
Сквозное выравнивание. Опция усреднения выравнивания элементов внутри разных групп.
Путь к данным заголовка. Если указано, то в скобках после названия группы это указывается.
Формат. Формат представлений чисел, дат и булева типа. - если указан путь к данным заголовка.
Использование ТекущейСтроки, ИспользуемаяТаблица. Для мобильного приложения.

Детали для элемента Строка.

Тип оформления может быть строкой ввода, подписью, переключателем, картинкой, HTML документом, PDF документом, текстовым документом.

Тип оформления Картинка Нужен для отображения сохраненной в базе в хранилище значений изображения.

Тип оформления Поле html документа. Позволяет загрузить html код, добавленный в виде текста. Может подгружать картинки из внешних источников (Интернета). Но грузился долго. Но если добавить в событие ПриОткрытии - остальная форма доступна.

```
МояСтрока = "<!DOCTYPE html><html lang=""en""><head><meta charset=""UTF-8""><meta
name=""viewport"" content=""width=device-width, initial-
scale=1.0""><title>Документ</title></head><body><h1>Привет!</h1><img
src=""https://helpf.pro/uploads/avatars/00.gif""></body></html>"
```

Переадресация страницы сработала.

```
МояСтрока = "<!DOCTYPE html><html><head><meta http-equiv=""refresh""
content=""0;url=https://dzen.ru""></head></html>"
```

Если ссылка открывается в новой странице - тогда в браузере. JS скрипт вывода сообщения пользователю не выполнил. А скрипт отображения часов сработал:

```
МояСтрока = "<!DOCTYPE html><html></head><body><div id=""clock""></div><script>window.onload =
function(){window.setInterval(function(){var now = new Date();var clock =
document.getElementById(""clock"");clock.innerHTML =
now.toLocaleTimeString();},1000);};</script></body></html>";
```

Поддержка стилей есть. Резюме: внешний html ресурс возможен, но стоит 2 раза подумать, выпить глоток кофе и еще раз подумать.

Создать html ссылку

Оказалось непростым делом. Аж уже начал думать, что через html документ делать - в топе поиска устаревшая информация. Создается команда формы,

```
&НаКлиенте
Процедура ОткрытьПомощьПинг(Команда)
ПерейтиПоНавигационнойСсылке("http://tvm16:81/doku.php/aisimushestvo/pingcheck");
КонецПроцедуры
```

перетаскивается на форму, и в свойствах кнопки выставляется вид Гиперссылка.

??????????????
?????????????? ? ??????????

Вывод информации пользователю

```
Предупреждение("Текст", Таймаут_число, "Заголовок окна сообщения"); //окно сообщения
Сообщить("1!", СтатусСообщения.Важное); //в блоке сообщений
```

Получение информации от пользователя во всплывающем окне

```
ВвестиЧисло();
ВвестиЗначение();
ВвестиСтроку();
ВвестиДату();
```

Вариации модальных окон (предыдущие примеры) использовать нежелательно, скорее всего от данной технологии будут отказываться. Лучше использование асинхронного взаимодействия. В частности, можно использовать процедуру “ПоказатьВопрос()” в сочетании с механизмом описания оповещения.

```
&НаКлиенте
Процедура СоздатьДоговоры(Команда)
Если ЭтотОбъект.Модифицированность Тогда
ОписаниеОповещения = Новый ОписаниеОповещения("СоздатьДоговорыЗавершение", ЭтотОбъект);
ПоказатьВопрос(ОписаниеОповещения, "Документ должен быть записан. Записать и продолжить?",
РежимДиалогаВопрос.ДаНет);
Иначе
//Если спрашивать не требуется вызываем процедуры завершения
СоздатьДоговорыЗавершение(КодВозвратаДиалога.Нет);
КонецЕсли;
КонецПроцедуры

&НаКлиенте
Процедура СоздатьДоговорыЗавершение(Результат, Параметры = Неопределено) Экспорт
Если Результат = КодВозвратаДиалога.Да Тогда
//Действия выполняемые при положительном ответе пользователя
КонецЕсли;
```

```
//Действия выполняемые независимо от ответа пользователя
КонецПроцедуры
```

Обновление данных на форме в пределах своей сессии

Оповещения

Статья про оповещения пользователю

Под термином "Оповещение" в одном контексте оповещения пользователю (ссылка выше), в другом - способ взаимодействия между формами в пределах одной сессии (это важно!). Для второго варианта в форме пишется обработчик оповещений, при генерации в форме-источнике запускается обработчик и в обработчике можно фильтровать события по источнику и названию. Типа шина межоконного клиентского взаимодействия в пределах одной сессии.

```
//В первой форме оповещаются все открытые формы
Оповестить("ИмяСобытия", Данные, "Форма1");


//В другой форме следует назначить обработчик события оповещения.
Процедура ОбработкаОповещения(Событие, Данные, Источник)
   Если Событие = "ИмяСобытия" И Источник = "Форма1" Тогда
   КонецЕсли;
КонецПроцедуры
```

И - нихера не заработало! А все потому, что процедуру ОбработкаОповещения нужно создавать не лапками, а через ПКМ на форме - События - ОбработкаОповещения либо указав созданную лапками процедуру в событии ОбработкаОповещения. И реально работает только если форма открыта.

Обновление данных на форме у всех пользователей

Здесь кроется одина из фундаментальных проблем 1С, понять и простить. При открытии формы редактирования (например элемента справочника, ... будем называть его объектом), до версии 8.3.14 объект блокировался на изменение у других пользователей. Т.е., если я открыл в справочнике Номенклатура объект Ручка красная, то если кто-то попытается его открыть у себя - получит сообщение о блокировке на запись и все. И пока я просто держу открытой эту вкладку- она заблокирована у всех остальных. После 8.3.14 стало чуть лучше, блокировка для данных открытой формы снимается через 1 минуту после сохранения. Работает на толстом клиенте в файловом режиме. Однако в web клиенте в файловом режиме блокировка устанавливается после первого нажатия Сохранить, затем не снимается до закрытия формы. Подробнее о механизме блокировок

Динамические списки: Элементы.ИмяДинамическогоСписка.Обновить()

Реквизиты формы: Методы ЭтаФорма.Прочитать() и ЭтаФорма.ОбновитьОтображениеДанных(). Метод Прочитать обновляет объект управляемой формы. Метод ОбновитьОтображениеДанных принудительно обновляет содержание элементов управления, особенно полезно, когда изменения значений реквизитов формы происходят вне самой формы. Это гарантирует, что каждый элемент в форме будет соответствовать последним изменениям данных в системе.

Динамическое обновление открытой формы. Методы в предыдущем разделе должны быть чем-то вызваны. Периодические задания исполняются только на сервере, для клиента не подходят. Шикарная статья про обработчики Понять и простить. С ностальгией вспомни школьные годы, метод обновления периодическими опросами сервера и вперед! Базовый код:

```
&НаКлиенте
Процедура ОболочкаОбработчика()
Элементы.СписокГородов.Обновить();
КонецПроцедуры

&НаКлиенте
Процедура ПодключитьОжидание(Команда)
ПодключитьОбработчикОжидания("ОболочкаОбработчика", 10, Ложь);
КонецПроцедуры

&НаКлиенте
Процедура ОтключитьОжидание(Команда)
ОтключитьОбработчикОжидания("ОболочкаОбработчика");
КонецПроцедуры
```

Это работает и в web клиенте. Не нашел способа получить список подключенных обработчиков, поэтому необходимы глазками-контролируемые правила создания и отключения обработчиков.

Важно: Если действие на сервере затратило меньше времени, чем таймаут, то обработчик строго по таймауту вызовет нужное, будто серверной приостановки и не было; а если больше, то "собьётся" и вызовется сразу по возвращении с сервера.

При высокой нагрузке на базу стоит усложнить алгоритм см. Динамические списки раздел Обновление данных.

Пример автоматического обновления данных в таблице значений.

Есть справочник Города, только Наименование. Есть обработка, где в таблице будем выводить и обновлять значения из справочника. Практически это может пригодиться только

если форма объемная, нужно поддерживать актуальность одной таблицы из нескольких, а полное обновление формы замедлит работу.

У элемента Таблица нужно добавить колонки реквизитов, в этом случае - одну колонку Наименование. При добавлении на форму, будет предложено добавить колонки.

Вроде не может использоваться на веб клиенте, но у меня поехало.

```
&НаКлиенте
Процедура ОболочкаОбработчика()
	//Элементы.СписокГородов.Обновить();
	ОбновитьТаблицу();
КонецПроцедуры

&НаКлиенте
Процедура ПодключитьОжидание(Команда)
	ПодключитьОбработчикОжидания("ОболочкаОбработчика", 10, Ложь);
КонецПроцедуры

&НаКлиенте
Процедура ОтключитьОжидание(Команда)
	ОтключитьОбработчикОжидания("ОболочкаОбработчика");
КонецПроцедуры

&НаСервере
Процедура ОбновитьТаблицу();
	Запрос = Новый Запрос;
	Запрос.Текст =
	"ВЫБРАТЬ
	|	Города.Наименование КАК Наименование
	|ИЗ
	|	Справочник.Города КАК Города";
	РезультатЗапроса = Запрос.Выполнить();
	//СпособОбхода = ОбходРезультатаЗапроса.Прямой;
	СписокГородовТаблица.Загрузить(РезультатЗапроса.Выгрузить())
КонецПроцедуры

&НаСервере
Процедура ПриСозданииНаСервере(Отказ, СтандартнаяОбработка)
	ОбновитьТаблицу();
КонецПроцедуры
```

Метод Загрузить(РезультатЗапроса.Выгрузить) работает если наименования колонок совпадают с наименованиями столбцов в запросе.

Заметил, что при обновлении динамического списка на указателе мыши возникает кружок ожидания, однако фокус в списке после обновления не смещается. У таблицы кружок не возникает, однако фокус смещается на первый элемент.

На потом, не завершено: Появилась идея в обновления таблицы только в случае изменения хэша. Нужно бы проверить скорость работы, но пока не до этого. Однако сам факт вычисления хэша затратен.

На потом, не завершено: Предположительный алгоритм обновления с динамическим изменением периода опроса.

Проблема: в случае большого количества клиентов со включенными обработчиками возможна нагрузка на сервер.

Задача: Необходим алгоритм, управляющий периодами обновления данных на клиентах с приоритетами по объектам. Также при необходимости (например, наличие ресурсоемких задач на сервере), должен быть способ общего увеличения периода. На сервере могут быть другие базы, иные серверные скрипты, поэтому у нас нет полного доступа ко всем процессам, только локальная оценка производительности. Все клиенты равнозначны (хотя для общего решения лучше делить по пользователям, но потом).

Решение:

На сервере:
Регистр сведений, измерения - НазваниеПроцедуры, Ресурсы - ВремяПоследнегоИзменения, ЗадержкаОбновления. Служит для серверного изменения при обновлении данных. Способ реализации обновления этого регистра - отдельный вопрос.
Зарезервированное название процедуры - ПериодОбновленияРегистра для управления обновлением регистра на клиенте. При нагрузке увеличить задержку обновления и общий период обновления.

На клиенте:
ПараметрСеанса Строка, сериализованный аналог полей регистра сведений об обновлениях, только индивидуально для каждого сеанса. Часть процедур, которые нужны конкретному сеансу. На формы добавить события ПриОткрытии/Закрытии, для поддержания актуальности о подписанных процедурах. Процедуры сериализации / десериализации данных и обновления регистра. При пустом параметре, обновления регистра прекращаются. Если ВремяПоследнегоИзменения в регистре больше чем в параметре сеанса, то через ЗадержкуОбновления произвести обновление. Т е у каждой формы практически одинаковый алгоритм проверки / обновления.

Вопросы к постановке задачи и алгоритму:

1. Как определить, что количество клиентов стало "большим"
2. А может быть это не нужно?

?????????? ??????

Используется для отображения информации из любых таблиц независимо от их типа.

Похоже, динамический список формируется на сервере и затем представление передается на клиент. Поэтому использование именно динамического списка, а не например таблицы значений, должно быть осознанно. Динамический список предоставляет встроенные опции (стандартные команды, переходы для выделенных объектов), однако может сильно тормозить работу в случаях например часто меняющейся выборки из большого объема данных. Иногда эффективнее один раз загрузить на клиент данные и организовать фильтрацию на клиенте, без постоянной нагрузки на сервер и каналы связи.

Шаги использования динамического списка:

1 Наполнение данными

Данные могут быть получены из основной таблицы или через произвольный запрос. Способ получения данных настраивается в свойствах динамического списка, галочка Произвольный запрос.

Для обоих вариантов существует настройка Отбор (какие данные попадут в итоговый список), результат его работы пересекается с настройками запроса, поэтому требует углубленного понимания. Скорее всего, такая херня возникла из-за попытки сделать сложную вещь доступной прогерам-мышкотыкам (как и все в 1С), но потом пришли к выводу, что потенциал у данного типа данных большой, реализовать все возможные варианты через удобный графический интерфейс нельзя, а часть интерфейса создана, пусть будет. Понять и простить.

	Основная таблица	**Произвольный запрос**

Набор данных	Поля одной таблицы	Поля любого запроса
Интерактивная настройка (через отбор)	Настройка в конфигураторе повторяет настройку отображения в пользовательском интерфейсе. Ручное указание конкретного параметра либо общие типы (пустое, больше/меньше, ...)	Нельзя изменить запрос, остальное так же
Программная настройка параметров, включая параметры из реквизитов	Нет	Есть
Встроенный функционал редактирования списка	Да	Только если указана опция "Основная таблица".

Можно сделать вывод: настройка динамического списка через отбор конфигуратора актуальна для простейших задач (убрать пустые, ограничить по фиксированному значению, ...). В остальных случаях эффективнее работать через фильтрацию в параметрах запроса. Поэтому настройку отбора через вкладки с получением данных из основной таблицы рассмотрим в разделе Настройка отображения, далее будем рассматривать фильтрацию через произвольный запрос.

Произвольный запрос. Выберем следующие данные:

```
ВЫБРАТЬ
	ДокументПоступлениеТоваров.Ссылка,
	ДокументПоступлениеТоваров.Номер,
	ДокументПоступлениеТоваров.Дата,
	ДокументПоступлениеТоваров.Фирма,
	ДокументПоступлениеТоваров.Филиал,
	ДокументПоступлениеТоваров.ПереченьТоваров
ИЗ
	Документ.ПоступлениеТоваров КАК ДокументПоступлениеТоваров
ГДЕ
    ДокументПоступлениеТоваров.Дата < &НужнаяДата
```

Для параметров необходимо задать значение перед использованием. Это настраивается в модуле формы в процедуре «ПриСозданииНаСервере».

```
&НаСервере
Процедура ПриСозданииНаСервере(Отказ, СтандартнаяОбработка)
	ДокументыПоступленийТоваров.Параметры.УстановитьЗначениеПараметра("НужнаяДата",
ТекущаяДата());
КонецПроцедуры
```

ДокументыПоступленийТоваров - имя реквизита Динамический список.

2 Настройка отображения

Для всех разделов.
Несколько правил объединяются по способу И, правила можно объединять в группы, а для групп настраивать способ И, ИЛИ, НЕ.
В рамках одного пользователя сохраняется последняя настроенная группа правил.
Можно создавать преднастроенные наборы правил.

Отбор. Задается фиксированная настройка динамического списка. Виды сравнения зависят от типа данных поля, бывают в частности фиксированные (равно, больше, меньше), в списке, заполнено, в группе и обратные операции.

Порядок. Задается поле и тип сортировки. Несколько правил применяются последовательно, в рамках одинакового первого правила сортируется по следующему и т д.

Условное оформление. Может быть несколько правил, для каждого правила параметры оформления, условия и оформляемые поля. Параметры оформления: цвет фона, цвет текста, шрифт, положение текста, выделение отрицательного, шаблон текста, видимость, доступность. Т е здесь, как и в правилах отбора, можно даже ограничить видимость элементов. Сравнение способов ограничения видимости:

	Запрос	Отбор	Условное оформление
Точка фильтрации	На сервере	На сервере	На клиенте
Простота	Сложно	Просто	Просто
Скорость создания фильтра	Медленно	Быстро	Быстро
Настраиваемость	Динамическая	Статическая	Статическая
Объем передаваемых данных	Только попавшие в фильтр	Только попавшие в фильтр	Все
Изменение пользователем	То что разрешено разработчиком	Да	Да

Группировка. Несколько правил применяются последовательно, в рамках одинакового первого правила группируется по следующему и т д.

3 Интерактивное взаимодействие

В случае установленной Основной таблицы доступен стандартный набор команд (создание, удаление, изменение).

```
Элементы.элСписок.ТекущаяСтрока = НоменклатураСсылка; //получение и изменение текущей строки,
Сообщить(Элементы.элСписок.ТекущаяСтрока.Цена);// данные заново получаются из базы
```

```
Сообщить(Элементы.элСписок.ТекущиеДанные.Цена);// данные получаются из существующих данных


//перебор строк динамического цикла
&НаСервере
Процедура ЦиклПоСтрокамДинамическогоСписка()
    ТЗ_ДС = СписокВТЗ();
    Для Каждого Стр Из ТЗ_ДС Цикл
        // .........
    КонецЦикла;
КонецПроцедуры
```

5 Обновление данных

Сама по себе информация в списках не обновляется. Необходимо два компонента: процедура, генерирующая информацию о факте изменения данных и способ распространения информации об изменении по открытым формам (метод обновления). Стандартного общего события при обновлении данных например у справочника не нашел. Поэтому для реализации задачи поддержки актуальности чувствительных данных в открытых формах возможны следующие варианты:

1. Отказаться от стандартного механизма работы с данными, переписать процедуры обновления в отдельные формы и добавить оповещение об обновлении (см. ниже). При небольшом количестве точек изменения это сработает, но поддержка муторная. И если кто-то из разработчиков случайно забудет о необходимости использования только специальных процедур и добавит случайно прямое изменение данных - появятся фантомные неактуальные данные. А еще - см. блок "Нае*али, но понять и простить"
2. Периодический контроль на уровне таблицы средствами языка 1С и оповещения в случае изменения. Упрощает задачу с точки зрения контроля точек изменения данных, но хорошо нагружает сервер. К тому же, это будет обновление с задержкой (минимум в минуту для файловой базы). Если таблицы объемные, данные меняются относительно нечасто, то постоянное хеширование сильно замедлит работу.
3. Вариации на тему генерации события изменения на уровне базы данных, дополнительных сервисов на чем-нибудь. Возможно с добавлением endpoint в 1С. Например, клиент-серверный вариант, БД Postgresql (для mssql похоже все гораздо сложнее). В 1С поднимается web сервис, при обращении на который генерируются оповещения по открытым клиентским формам. На postgresql устанавливается расширение plpython3u, настраивается триггер на изменение нужных таблиц и за счет этого генерируется REST запрос при изменении данных. Решение на случай "ПИЗДЕЦ КАК НАДО СОБЫТИЙ ДОХЕРА В РАЗНЫХ ТОЧКАХ ПОЛЬЗЫ НЕ МОГУТ САМИ ПОСТОЯННО ОБНОВЛЯТЬ, НА СЕРВЕР НЕТ ДЕНЕГ ЗАТО ЕСТЬ ДЕНЬГИ НА РАЗРАБОВ". Решение сложно сопровождаемое, зависящее от имен нужных таблиц, требующее разработчиков на других языках. Очень дорогое.

4. Периодический опрос сервера из формы. Нагрузка растет произведением от количества клиентов, количества контролируемых таблиц, их объема и минимального периода обновления. Если добавить возможности периодического регистра сведений, то нагрузка перестанет зависеть от количества клиентов. Возможно эту задачу вынести на отдельный сервер, организовав например кластер БД, скрипт по пересчету на другом сервере и еще какое-нибудь извращение. Но все равно обновление с задержкой, от 1 минимального периода обновления. Самый допотопный способ, но при некоторых условиях имеет право на использование. Дополнение: когда это писал, реально думал что это древняя технология и нужно просто узнать, как это реализуется в 1С. Но... Похоже это единственный рабочий способ,
5. Есть отдельный продукт 1С:Предприятие - Сервер взаимодействия, но нужно тестировать.

Но в целом, 1С даже близко не позиционируется как система реального времени / мгновенного обновления / совместного редактирования, так что особых претензий к отсутствию данных технологий быть не может. Возможно, стоит пересмотреть формулировку задачи или вынести данный функционал совсем за пределы 1С.

В перечисленных способах используется термин "Оповещение об обновлении". Методы оповещения представлены ниже. Метод ОповеститьОбИзменении работает только для динамических списков, да и то только для списков с указанной основной таблицей. Метод Оповестить универсальный.

Здесь возникает парадокс: обычно начинает использоваться дополнительная технология в связи с упрощением решения нужной задачи. Однако если эта технология ведет к дополнительным трудностям, то стоит сопоставить получаемые преимущества, возникающее усложнение и степень решения задачи.

Методы обновления

стр. 576 книги Разработка интерфейсов.

Метод ОповеститьОбИзменении() В него передается ссылка на объект (или ключ записи), об изменении которого нужно оповестить формы. Он уведомит все динамические списки, расположенные в созданных на клиенте формах, об изменении этого объекта, и они обновят свои данные. Но этот метод не обновит те динамические списки, у которых не задана основная таблица. Преимущество: процесс автоматический, без изменения форм. Но этот метод можно использовать только из клиента. Как следствие, списки в другой сессии / у другого пользователя не обновляются.

```
//ДобавитьЭлементНаСервере() - функция, возвращающая ссылку на новый элемент
//в какой-то форме
СсылкаНаНовыйЭлемент = ДобавитьЭлементНаСервере();
ОповеститьОбИзменении(СсылкаНаНовыйЭлемент);
```

Метод Оповестить() Нужно добавить код обработки оповещений в формах, в которых нужно что-то обновлять. Метод
глобального контекста Оповестить() отсылает оповещение всем созданным (не обязательно открытым) формам. В форме в обработчике события ОбработкаОповещения() можно обработать это сообщение и выполнить нужную модификацию формы.

И один хер у другого пользователя не обновляется!

```
&НаКлиенте
Процедура ОбщееОповещение(Команда)
	Элем = СоздатьСОповещениемНаСервере();
	Оповестить("ДобавлениеЭлемента");
КонецПроцедуры

&НаКлиенте
Процедура ОбработкаОповещения(ИмяСобытия, Параметр, Источник)
	Если ИмяСобытия = "ДобавлениеЭлемента" Тогда
		Элементы.ИзСправочника.Обновить();
	КонецЕсли;
КонецПроцедуры
```

Нае*али, но понять и простить.

У 1С нет стандартных технологии а-ля longpool, только метод Обновить.
