Оглавление

[ВАЖНЫЕ ССЫЛКИ 3](#_Toc167011795)

[Форум программиста 1с, много полезного: 3](#_Toc167011796)

[Универсальные функции ЗУП 3.1 / ЗКГУ 3.1, которые помогут в разработке 3](#_Toc167011797)

[БСП 3](#_Toc167011798)

[Шаблоны текста 3](#_Toc167011799)

[Консоль Запросов 3](#_Toc167011800)

[СКД программно 3](#_Toc167011801)

[Получаю переменную "ИмяПеременной" путем сложения строк: Строка(Имя) + "Переменной" - получается строковое значение "ИмяПеременной". В модуле есть переменная с таким же именем, можно ли путем сложения строк Строка(Имя) + "Переменной" получить не строку а переменную? 3](#_Toc167011802)

[ДвиженияДокумента отчет 3](#_Toc167011803)

[Сервис 1С: 4](#_Toc167011804)

[Подключение к хранилищу: 4](#_Toc167011805)

[Нет свободной лицензии: 4](#_Toc167011806)

[Инструкция Хранилище 1с:Университет: 4](#_Toc167011807)

[Неверный формат хранилища данных 'file://C:/Users/anna/AppData/Local/1C/1cv8/b6a310d6-972a-45ce-a21f-79e1b2590868/Config/RuntimeCacheStorage" 5](#_Toc167011808)

[Очистка кеша 1С 5](#_Toc167011809)

[Отключить 1с отчетность 5](#_Toc167011810)

[Горячие клавиши: 6](#_Toc167011811)

[Запросы 6](#_Toc167011812)

[Значение перечисления в запросе: 6](#_Toc167011813)

[Тип регистратора в запросе: 6](#_Toc167011814)

[Результат выполнения запроса: 6](#_Toc167011815)

[Функции запроса (сумма, кол-во, кол-во различные и т.д.): 6](#_Toc167011816)

[Условие в запросе задать параметром: 6](#_Toc167011817)

[В ЗАПРОСЕ НАЗВАНИЕ МЕСЯЦА СТРОКОЙ 6](#_Toc167011818)

[ПОЛЕЗНОЕ 7](#_Toc167011819)

[Предопределенное значение 1с на клиенте 7](#_Toc167011820)

[Двойной клик для выбора 7](#_Toc167011821)

[ФормаПодбора: 7](#_Toc167011822)

[ФормаОсновная: 7](#_Toc167011823)

[ОГРАНИЧИТЬ СПИСОК ВЫБОРА 7](#_Toc167011824)

[НачисленияВидРасчетаНачалоВыбора 9](#_Toc167011825)

[НачисленияВидРасчетаНачалоВыбораИзСписка 9](#_Toc167011826)

[Получить выбранные элементы с формы выбора 9](#_Toc167011827)

[ПРОВЕРКИ 9](#_Toc167011828)

[Проверка ТЧ документа на дубли строк 9](#_Toc167011829)

[Работа со строками 11](#_Toc167011830)

[Удалить пробелы 11](#_Toc167011831)

[Первая буква заглавная 11](#_Toc167011832)

[ИнициалыФамилия 11](#_Toc167011833)

[ДАТЫ 14](#_Toc167011834)

[Добавить или убавить от даты несколько месяцев 14](#_Toc167011835)

[ТАБЛИЦА ЗНАЧЕНИЙ 14](#_Toc167011836)

[Отбор в ТЗ 14](#_Toc167011837)

[Отбор структура в ТЗ + ТЗ.Скопировать 14](#_Toc167011838)

[18. Простой трюк для быстрого объединения таблиц значений 14](#_Toc167011839)

[Работа со строками ТЗ 14](#_Toc167011840)

[Заполнить одну строку другой: 14](#_Toc167011841)

[ТабличныйДокумент 14](#_Toc167011842)

[Проверка на заполненность табличного документа 14](#_Toc167011843)

[Описание типов 14](#_Toc167011844)

[Дерево в ТЗ 16](#_Toc167011845)

[ТЗ в Массив 16](#_Toc167011846)

[Двоичные данные 18](#_Toc167011847)

[Из двоичных данных в ТабДок 18](#_Toc167011848)

[Двоичные данные в ПДФ 18](#_Toc167011849)

[СООБЩЕНИЯ 18](#_Toc167011850)

[Вопрос РежимДиалогаВопрос.ДаНет 18](#_Toc167011851)

[СообщениеПользователю 18](#_Toc167011852)

[БСП 18](#_Toc167011853)

[СообщениеПользователю 19](#_Toc167011854)

# ВАЖНЫЕ ССЫЛКИ

## Форум программиста 1с, много полезного:

<https://almaz-sharipov.ru/catalog/1c-develop>

## Универсальные функции ЗУП 3.1 / ЗКГУ 3.1, которые помогут в разработке

<https://infostart.ru/1c/articles/942569/>

## БСП

[**https://infostart.ru/1c/articles/1398340/**](https://infostart.ru/1c/articles/1398340/)

## Шаблоны текста

[**https://koder.by/shablony\_avtozameny\_1s.php**](https://koder.by/shablony_avtozameny_1s.php)

## Консоль Запросов

[**https://infostart.ru/1c/articles/1828459/**](https://infostart.ru/1c/articles/1828459/)

## СКД программно

**Ссылки**:

<https://infostart.ru/1c/articles/1185890/>

<https://1code.info/1s-8-3-skd-rabota-s-programmnym-otborom-gruppirovkoj-naborom-dannyh-polzovatelskimi-polyami/>

## Получаю переменную "ИмяПеременной" путем сложения строк: Строка(Имя) + "Переменной" - получается строковое значение "ИмяПеременной". В модуле есть переменная с таким же именем, можно ли путем сложения строк Строка(Имя) + "Переменной" получить не строку а переменную?

**Ссылка**: <https://forum.mista.ru/topic.php?id=470526>

Переменная = Вычислить("Имя");

Вычислить("СтруктураПараметровЗаявления." + СвойстваОбъекта[0].Заголовок)

**Выполнить (Execute)**

**Синтаксис:**Выполнить(<Строка>)

**Параметры:**  
<Строка> - Строка, содержащая текст исполняемого кода.

**Описание:**Позволяет выполнить фрагмент кода, который передается ему в качестве строкового значения.

## ДвиженияДокумента отчет

<https://oparin.info/1c/otchet-dvizheniyadokumenta-dlya-svoego-dokumenta/>

<https://its.1c.ru/db/bsp319doc#content:1520:hdoc>

# Сервис 1С:

## Подключение к хранилищу:

Для подключения к хранилищу: нужно сначала зайти через мой компьютер:

\\univer.ustu\storage1C\eldoc

Exchange

zkgu!2020!902\_slOvo

## Нет свободной лицензии:

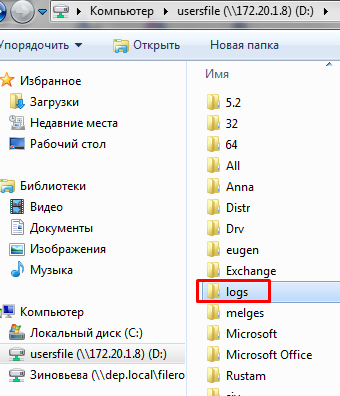
C:\Program Files\1cv8\conf\nethasp

NH\_SERVER\_ADDR = 172.20.1.30, 172.20.1.8 (ИЛИ НАОБОРОТ)

## Инструкция Хранилище 1с:Университет:

1. Развернуть базу

2. Создать папку из ошибки (на д папка логов)



3. Отключиться от хранилища

4. Подключиться к хранилищу

5. Перезапустить конфигуратор

6. Захватить корень **нерекурсивно**

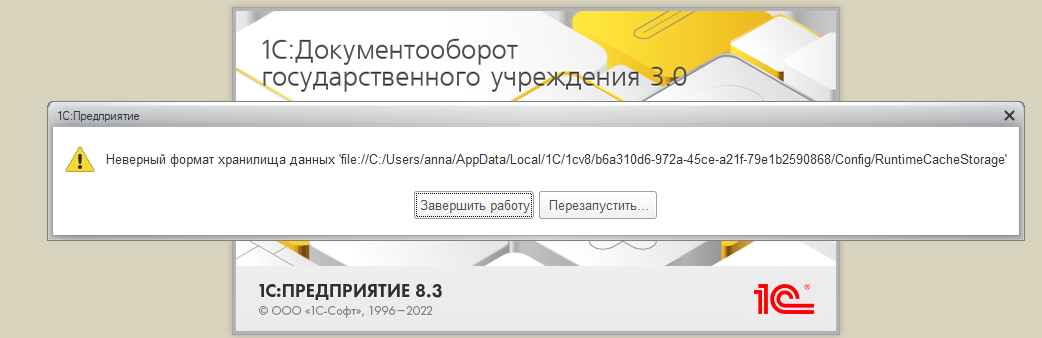
7. ктрл с ктрл в

8. обновить базу

9. отладить

10. поместить в хранилище **рекурсивно**

## Неверный формат хранилища данных 'file://C:/Users/anna/AppData/Local/1C/1cv8/b6a310d6-972a-45ce-a21f-79e1b2590868/Config/RuntimeCacheStorage"

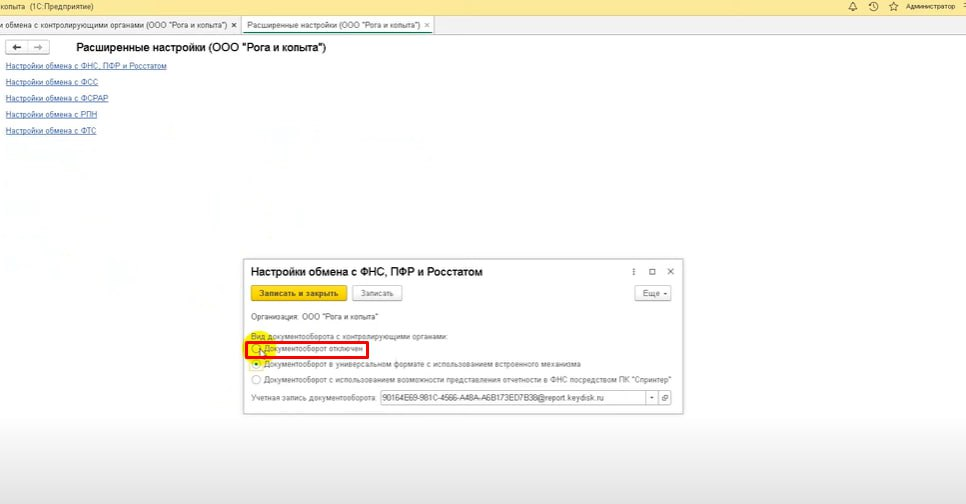
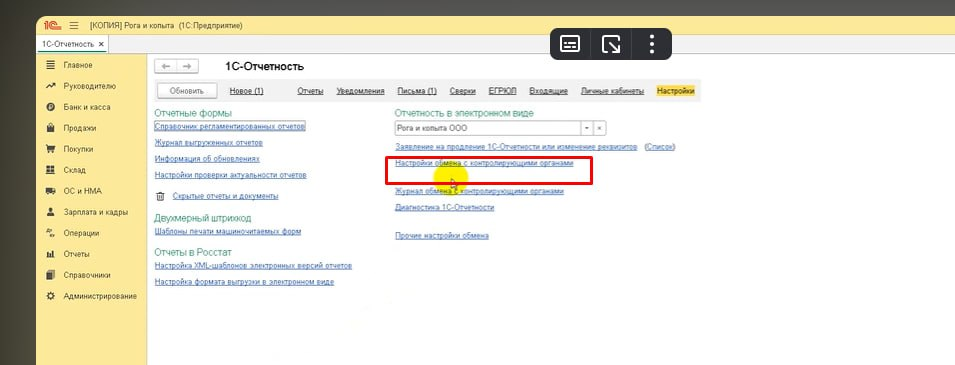


## Очистка кеша 1С

Информацию, как почистить локальный кеш, Вы без труда найдете в интернете. Об этом много уже написано (в том числе и в нашей статье: [Как очистить кэш в 1С 8.3](https://programmist1s.ru/kak-pochistit-kesh-v-1s/)), и повторяться я не буду.

Порой достаточно удалить строку с наименованием информационной базы из списка, а затем снова добавить. При этом создастся новая, чистая папка для кеша. Часто таким способом пользоваться не рекомендую, так как папка со старым кешем остается и засоряет диск.

## Отключить 1с отчетность



# Горячие клавиши:

1. ctl + пробел = посмотреть далее через точку

2. ctrl + alt + f = отформатировать

3. ctrl + Q = вывести быстро шаблон кода

4. Ctrl + Shift + N = область макета (шапка и тд)

# Запросы

## Значение перечисления в запросе:

ВТ\_Жильцы.Причина=ЗНАЧЕНИЕ(Перечисление.УлГТУ\_ОбщежитиеПричинаЗаселения.ОбучениеБюджет)";

## Тип регистратора в запросе:

|ГДЕ

| УлГТУ\_ОбщежитиеКомнаты.Регистратор ССЫЛКА Документ.УлГТУ\_ОбщежитиеВвод

## Результат выполнения запроса:

Выборка:

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

ТЗ:

ТЗ = Запрос.Выполнить().Выгрузить();

Пакеты запросов:

МассивРезультатов = Запрос.ВыполнитьПакет();

тзРезультатЗапроса = МассивРезультатов[<ИндексЗапроса>].Выгрузить();

// ИЛИ:

Выборка = МассивРезультатов[0].Выбрать();

ИндексЗапроса с 0!!!

## Функции запроса (сумма, кол-во, кол-во различные и т.д.):

<https://helpme1s.ru/yazyk-zaprosov-1s-8-3-dlya-nachinayushhix-programmistov-gruppirovka>

**ЕСТЬNULL**(<Проверяемое выражение>, <Выражение замены>):

ЕСТЬNULL(УчетНоменклатурыОстатки.КоличествоОстаток, 0) КАК КоличествоОстаток

## Условие в запросе задать параметром:

Запрос.Текст = СтрЗаменить(Запрос.Текст, "ГДЕ &Усл"," ");

## В ЗАПРОСЕ НАЗВАНИЕ МЕСЯЦА СТРОКОЙ

Выбор

Когда МЕСЯЦ(ОсновныеНачисления.ПериодРегистрации)=1 Тогда "Январь"

Когда МЕСЯЦ(ОсновныеНачисления.ПериодРегистрации)=2 Тогда "Февраль"

Когда МЕСЯЦ(ОсновныеНачисления.ПериодРегистрации)=3 Тогда "Март"

Когда МЕСЯЦ(ОсновныеНачисления.ПериодРегистрации)=4 Тогда "Апрель"

Когда МЕСЯЦ(ОсновныеНачисления.ПериодРегистрации)=5 Тогда "Май"

Когда МЕСЯЦ(ОсновныеНачисления.ПериодРегистрации)=6 Тогда "Июнь"

Когда МЕСЯЦ(ОсновныеНачисления.ПериодРегистрации)=7 Тогда "Июль"

Когда МЕСЯЦ(ОсновныеНачисления.ПериодРегистрации)=8 Тогда "Август"

Когда МЕСЯЦ(ОсновныеНачисления.ПериодРегистрации)=9 Тогда "Сентябрь"

Когда МЕСЯЦ(ОсновныеНачисления.ПериодРегистрации)=10 Тогда "Октябрь"

Когда МЕСЯЦ(ОсновныеНачисления.ПериодРегистрации)=11 Тогда "Ноябрь"

Когда МЕСЯЦ(ОсновныеНачисления.ПериодРегистрации)=12 Тогда "Декабрь"

Конец Как НазваниеМесяца

# ПОЛЕЗНОЕ

## Предопределенное значение 1с на клиенте

ПредопределенноеЗначение("Перечисление.ЮридическоеФизическоеЛицо.ЮридическоеЛицо");

## Двойной клик для выбора

&НаКлиенте

Процедура ПартииВыбор(Элемент, ВыбраннаяСтрока, Поле, СтандартнаяОбработка)

СтандартнаяОбработка = Ложь;

//ТекСтр = Элементы.Партии.ТекущиеДанные;

////ОповеститьОВыборе(Объект.Списание);

//НоваяСтрока = Объект.Списание.Добавить();

////НоваяСтрока = ТекСтр;

//Стр = Объект.Партии[ВыбраннаяСтрока];

//НоваяСтрока. = Стр;

////Объект.Списание.Добавить(ВыбраннаяСтрока);

ЗаполнитьЗначенияСвойств(Объект.Списание.Добавить(), Объект.Партии[ВыбраннаяСтрока]);

КонецПроцедуры

## ФормаПодбора:

&НаКлиенте

Процедура ЗавершитьПодборИПередатьДанныеВОсновнуюформу(Команда)

АдресВХ = Неопределено;

ЗавершитьПодборИПередатьДанныеВОсновнуюформуНаСервере(АдресВХ);

ВладелецФормы.ОбработатьПодбор(АдресВХ);

Закрыть();

КонецПроцедуры

&НаСервере

Процедура ЗавершитьПодборИПередатьДанныеВОсновнуюформуНаСервере(АдресВХ)

ПартииДляСписания = Партии.Выгрузить();

АдресВХ = ПоместитьВоВременноеХранилище(ПартииДляСписания, Новый УникальныйИдентификатор);

КонецПроцедуры

## ФормаОсновная:

&НаСервере

Процедура ОбработатьПодбор(АдресВХ) Экспорт

СписокПартий = ОбработатьПодборНаСервере(АдресВХ);

Объект.Списание.Загрузить(СписокПартий);

КонецПроцедуры

&НаСервере

Функция ОбработатьПодборНаСервере(АдресВХ)

Возврат ПолучитьИзВременногоХранилища(АдресВХ);

КонецФункции

&НаСервере

Процедура ОбработатьПодбор(АдресВХ) Экспорт

СписокПутевок = ОбработатьПодборНаСервере(АдресВХ);

Если СписокПутевок.Количество() <> 0 Тогда

Путевка = СписокПутевок[0];

СтрокаТекущая = Объект.Заявки.НайтиПоИдентификатору(Элементы.Заявки.ТекущаяСтрока);

СтрокаТекущая.Комната = Путевка.Комната;

СтрокаТекущая.ЦенаПутевки = Путевка.ЦенаПутевки;

КонецЕсли;

КонецПроцедуры

## ОГРАНИЧИТЬ СПИСОК ВЫБОРА

<http://1c-kod.ru/topic/6>

&НаСервере

Функция ПолучитьДоступныйМассивЗначений(Ученик)

Массив = Новый Массив;

Запрос = Новый Запрос;

Запрос.Текст =

"ВЫБРАТЬ

| УченикиРасчетныеСчета.РасчётныйСчёт КАК РасчётныйСчёт

|ИЗ

| Справочник.Ученики.РасчетныеСчета КАК УченикиРасчетныеСчета

|ГДЕ

| УченикиРасчетныеСчета.Ссылка = &Ученик";

Запрос.УстановитьПараметр("Ученик", Ученик);

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

Пока Выборка.Следующий() Цикл

Массив.Добавить(Выборка.РасчётныйСчёт);

КонецЦикла;

Возврат Массив;

КонецФункции

&НаСервере

Функция ПолучитьДоступныйСписокЗначений(Ученик)

СписокЗначений = Новый СписокЗначений;

Запрос = Новый Запрос;

Запрос.Текст =

"ВЫБРАТЬ

| УченикиРасчетныеСчета.РасчётныйСчёт КАК РасчётныйСчёт

|ИЗ

| Справочник.Ученики.РасчетныеСчета КАК УченикиРасчетныеСчета

|ГДЕ

| УченикиРасчетныеСчета.Ссылка = &Ученик";

Запрос.УстановитьПараметр("Ученик", Ученик);

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

Пока Выборка.Следующий() Цикл

СписокЗначений.Добавить(Выборка.РасчётныйСчёт);

КонецЦикла;

Возврат СписокЗначений;

КонецФункции

&НаСервере

Процедура УченикиРасчётныйСчётПриИзмененииНаСервере(Ученик)

СписокЗначений = ПолучитьДоступныйСписокЗначений(Ученик);

КонецПроцедуры

//&НаКлиенте

//Процедура УченикиРасчётныйСчётПриИзменении(Элемент)

// СтрокаТЧ = Элементы.Ученики.ТекущиеДанные;

// СписокЗначений = ПолучитьДоступныйСписокЗначений(СтрокаТЧ.Ученик);

// НайденныйЭлемент = СписокЗначений.НайтиПоЗначению(СтрокаТЧ.РасчётныйСчёт);

// Если НайденныйЭлемент = Неопределено Тогда

// Сообщить("Недоступный расчетный счет у ученика");

// КонецЕсли;

//КонецПроцедуры

//&НаСервере

//Функция ПолучитьФормуВыбора()

//

// Возврат Справочники.Ученики.ПолучитьформуВыбора();

//КонецФункции

&НаКлиенте

**Процедура УченикиРасчётныйСчётНачалоВыбора(Элемент, ДанныеВыбора, СтандартнаяОбработка)**

СтандартнаяОбработка = Ложь;

СтрокаТЧ = Элементы.Ученики.ТекущиеДанные;

УченикиРасчётныйСчётНачалоВыбораНаСервере(ДанныеВыбора, СтрокаТЧ.Ученик);

**КонецПроцедуры**

&НаСервере

**Процедура УченикиРасчётныйСчётНачалоВыбораНаСервере(ДанныеВыбора, Ученик)**

ДанныеВыбора = ПолучитьДоступныйСписокЗначений(Ученик);

**КонецПроцедуры**

НачисленияВидРасчетаНачалоВыбора

&НаКлиенте

**Процедура НачисленияВидРасчетаНачалоВыбора(Элемент, ДанныеВыбора, СтандартнаяОбработка)**

СтандартнаяОбработка = Ложь;

ПараметрыФормы = Новый Структура;

ЗначениеОтбора = Новый Структура("Наименование", Строка(ПредопределенноеЗначение("ПланВидовРасчета.ОсновныеНачисления.Больничный")));

ПараметрыФормы.Вставить("Отбор", ЗначениеОтбора);

ОткрытьФорму("ПланВидовРасчета.ОсновныеНачисления.ФормаВыбора", ПараметрыФормы, Элемент);

**КонецПроцедуры**

НачисленияВидРасчетаНачалоВыбораИзСписка

&НаКлиенте

**Процедура НачисленияВидРасчетаНачалоВыбораИзСписка(Элемент, СтандартнаяОбработка)**

Список = Новый СписокЗначений;

Список.Добавить(ПредопределенноеЗначение("ПланВидовРасчета.ОсновныеНачисления.Больничный"));

Результат = ВыбратьИзСписка(Список, Элемент);

СтандартнаяОбработка = Ложь;

**КонецПроцедуры**

## Получить выбранные элементы с формы выбора

Результат = ПолучитьФорму("Справочник.ТорговыеТочки.ФормаВыбора").ОткрытьМодально();

# ПРОВЕРКИ

## Проверка ТЧ документа на дубли строк

// Проверяет тч документа на дубли

//

// ДокументСсылка - ссылка на проверяемый документ

// ИмяТабЧасти - имя проверяемой табличной части (строка)

// ПроверяемРеквизиты - перечень проверяемых реквизитов (строка, реквизиты разделяются запятой)

**Процедура ПроверитьДублиСтрок(ДокументСсылка, ИмяТабЧасти, ПроверяемРеквизиты, Отказ) Экспорт**

ТаблДок = ДокументСсылка[ИмяТабЧасти].Выгрузить();

ТаблДок.Колонки.Добавить("КоличПроверкаДублей");

ТаблДок.ЗаполнитьЗначения(1, "КоличПроверкаДублей");

ТаблДок.Свернуть(ПроверяемРеквизиты,"КоличПроверкаДублей");

Для Каждого ТекСтрока Из ТаблДок Цикл

Если ТекСтрока.КоличПроверкаДублей > 1 Тогда

ДляСообщения = "";

СтрРекв = СокрЛП(ПроверяемРеквизиты);

Пока Найти(СтрРекв,",") > 0 Цикл

НаимРеквизита = СокрЛП(Лев(СтрРекв,Найти(СтрРекв,",") - 1));

СтрРекв = Прав(СтрРекв,СтрДлина(СтрРекв) - Найти(СтрРекв,","));

ДляСообщения = ДляСообщения+ТекСтрока[НаимРеквизита]+", ";

КонецЦикла;

ДляСообщения = ДляСообщения+ТекСтрока[СокрЛП(СтрРекв)];

Сообщение = Новый СообщениеПользователю;

Сообщение.Текст = "Проверьте одинаковые значения: " + ДляСообщения + ", сохранение невозможно";

Сообщение.Сообщить();

Отказ = Истина;

КонецЕсли;

КонецЦикла;

**КонецПроцедуры**

**&НаСервере**

**Процедура ОбработкаПроверкиЗаполненияНаСервере(Отказ, ПроверяемыеРеквизиты)**

//Если Объект.Приемы.Количество() = 0 Тогда

// Возврат;

//КонецЕсли;

//

//УлГТУ\_ОбщийМодуль.ПроверитьДублиСтрок(Объект.Ссылка, "Приемы", "ПриемНаРаботу", Отказ);

**КонецПроцедуры**

# Работа со строками

## Удалить пробелы

**СтрЗаменить(ТвояСтрока,Символы.НПП,"")**

### Первая буква заглавная

ОбластьИсполнителиИспользуемый.Параметры.Должность = **ВРег(Лев(СтрокаИсполнитель.Должность,1)) + Нрег(Сред(СтрокаИсполнитель.Должность,2));**

### ИнициалыФамилия

ОбластьИсполнителиИспользуемый.Параметры.ФИО = **ФизическиеЛицаКлиентСервер.ИнициалыФамилия(СтрокаИсполнитель.ФИО);**

///////////////////////////////////////////////////////////////////////////////////////////////////////

// Copyright (c) 2022, ООО 1С-Софт

// Все права защищены. Эта программа и сопроводительные материалы предоставляются

// в соответствии с условиями лицензии Attribution 4.0 International (CC BY 4.0)

// Текст лицензии доступен по ссылке:

// https://creativecommons.org/licenses/by/4.0/legalcode

///////////////////////////////////////////////////////////////////////////////////////////////////////

#Область ПрограммныйИнтерфейс

// Раскладывает полное имя физического лица на составные части - фамилию, имя и отчество.

// Если в конце полного имени встречаются "оглы", "улы", "уулу", "кызы" или "гызы",

// то они также считаются частью отчества.

//

// Параметры:

// ФамилияИмяОтчество - Строка - полное имя в виде "Фамилия Имя Отчество".

//

// Возвращаемое значение:

// Структура:

// \* Фамилия - Строка - фамилия;

// \* Имя - Строка - имя;

// \* Отчество - Строка - отчество.

//

// Пример:

// 1. ФизическиеЛицаКлиентСервер.ЧастиИмени("Иванов Иван Иванович")

// вернет структуру со значениями свойств: "Иванов", "Иван", "Иванович".

// 2. ФизическиеЛицаКлиентСервер.ЧастиИмени("Смит Джон")

// вернет структуру со значениями свойств: "Смит", "Джон", "".

// 3. ФизическиеЛицаКлиентСервер.ЧастиИмени("Алиев Ахмед Октай оглы")

// вернет структуру со значениями свойств: "Алиев", "Ахмед", "Октай оглы".

//

Функция ЧастиИмени(ФамилияИмяОтчество) Экспорт

Результат = Новый Структура("Фамилия,Имя,Отчество");

ЧастиИмени = СтрРазделить(ФамилияИмяОтчество, " ", Ложь);

Если ЧастиИмени.Количество() >= 1 Тогда

Результат.Фамилия = ЧастиИмени[0];

КонецЕсли;

Если ЧастиИмени.Количество() >= 2 Тогда

Результат.Имя = ЧастиИмени[1];

КонецЕсли;

Если ЧастиИмени.Количество() >= 3 Тогда

Результат.Отчество = ЧастиИмени[2];

КонецЕсли;

Если ЧастиИмени.Количество() > 3 Тогда

ДополнительныеЧастиОтчества = Новый Массив;

ДополнительныеЧастиОтчества.Добавить(НСтр("ru = 'оглы'"));

ДополнительныеЧастиОтчества.Добавить(НСтр("ru = 'улы'"));

ДополнительныеЧастиОтчества.Добавить(НСтр("ru = 'уулу'"));

ДополнительныеЧастиОтчества.Добавить(НСтр("ru = 'кызы'"));

ДополнительныеЧастиОтчества.Добавить(НСтр("ru = 'гызы'"));

ДополнительныеЧастиОтчества.Добавить(НСтр("ru = 'угли'"));

Если ДополнительныеЧастиОтчества.Найти(НРег(ЧастиИмени[3])) <> Неопределено Тогда

Результат.Отчество = Результат.Отчество + " " + ЧастиИмени[3];

КонецЕсли;

КонецЕсли;

Возврат Результат;

КонецФункции

// Формирует краткое представление из полного имени физического лица.

//

// Параметры:

// ФамилияИмяОтчество - Строка - полное имя в виде "Фамилия Имя Отчество";

// - Структура:

// \* Фамилия - Строка - фамилия;

// \* Имя - Строка - имя;

// \* Отчество - Строка - отчество.

//

// Возвращаемое значение:

// Строка - фамилия и инициалы. Например, "Пупкин В. И.".

//

// Пример:

// Результат = ФизическиеЛицаКлиентСервер.ФамилияИнициалы("Пупкин Василий Иванович");

// - возвращает "Пупкин В. И.".

//

Функция ФамилияИнициалы(Знач ФамилияИмяОтчество) Экспорт

Если ТипЗнч(ФамилияИмяОтчество) = Тип("Строка") Тогда

ФамилияИмяОтчество = ЧастиИмени(ФамилияИмяОтчество);

КонецЕсли;

Фамилия = ФамилияИмяОтчество.Фамилия;

Имя = ФамилияИмяОтчество.Имя;

Отчество = ФамилияИмяОтчество.Отчество;

Если ПустаяСтрока(Имя) Тогда

Возврат Фамилия;

КонецЕсли;

Если ПустаяСтрока(Отчество) Тогда

Возврат СтроковыеФункцииКлиентСервер.ПодставитьПараметрыВСтроку("%1 %2.", Фамилия, Лев(Имя, 1));

КонецЕсли;

Возврат СтроковыеФункцииКлиентСервер.ПодставитьПараметрыВСтроку("%1 %2. %3.", Фамилия, Лев(Имя, 1), Лев(Отчество, 1));

КонецФункции

// Проверяет, верно ли написано ФИО физического лица.

// ФИО считается верным, если содержит только кириллицу, либо только латиницу.

//

// Параметры:

// ФИО - Строка - фамилия, имя и отчество. Например, "Пупкин Василий Иванович".

// ТолькоКириллица - Булево - при проверке допустимой будет только кириллица в ФИО.

//

// Возвращаемое значение:

// Булево - Истина, если ФИО написано верно.

//

Функция ФИОНаписаноВерно(Знач ФИО, ТолькоКириллица = Ложь) Экспорт

ДопустимыеСимволы = "-";

Возврат (Не ТолькоКириллица И СтроковыеФункцииКлиентСервер.ТолькоЛатиницаВСтроке(ФИО, Ложь, ДопустимыеСимволы))

Или СтроковыеФункцииКлиентСерверРФ.ТолькоКириллицаВСтроке(ФИО, Ложь, ДопустимыеСимволы);

КонецФункции

#КонецОбласти

//Строева

// Формирует краткое представление из полного имени физического лица.

//

// Параметры:

// ФамилияИмяОтчество - Строка - полное имя в виде "Фамилия Имя Отчество";

// - Структура - части полного имени:

// \* Фамилия - Строка - фамилия;

// \* Имя - Строка - имя;

// \* Отчество - Строка - отчество.

//

// Возвращаемое значение:

// Строка - фамилия и инициалы. Например, "Пупкин В. И.".

//

// Пример:

// Результат = ФизическиеЛицаКлиентСервер.ИнициалыФамилия("Пупкин Василий Иванович");

// - возвращает "В.И. Пупкин".

**Функция ИнициалыФамилия(Знач ФамилияИмяОтчество) Экспорт**

Если ТипЗнч(ФамилияИмяОтчество) = Тип("Строка") Тогда

ФамилияИмяОтчество = ЧастиИмени(ФамилияИмяОтчество);

КонецЕсли;

Фамилия = ФамилияИмяОтчество.Фамилия;

Имя = ФамилияИмяОтчество.Имя;

Отчество = ФамилияИмяОтчество.Отчество;

Если ПустаяСтрока(Имя) Тогда

Возврат Фамилия;

КонецЕсли;

Если ПустаяСтрока(Отчество) Тогда

Возврат СтроковыеФункцииКлиентСервер.ПодставитьПараметрыВСтроку("%1.%2", Лев(Имя, 1), Фамилия);

КонецЕсли;

Возврат СтроковыеФункцииКлиентСервер.ПодставитьПараметрыВСтроку("%1.%2.%3", Лев(Имя, 1), Лев(Отчество, 1), Фамилия);

**КонецФункции**

//КонецСтроева

# ДАТЫ

## Добавить или убавить от даты несколько месяцев

ДобавитьМесяц(Начало, -3) – три месяца назад

# ТАБЛИЦА ЗНАЧЕНИЙ

## Отбор в ТЗ

<https://1c-programmer-blog.ru/programmirovanie/otbor-v-tablice-znachenij.html>

## Отбор структура в ТЗ + ТЗ.Скопировать

ОтборНужноСоздать = Новый Структура();

ОтборНужноСоздать.Вставить("НужноСоздать", Истина);

ОтборНужноСоздать.Вставить("НужноОбновить", Истина);

ОтборНужноСоздать.Вставить("Записывать", Истина);

ОтборНужноСоздать.Вставить("СсылкаДГУ", Справочники.Сотрудники.ПустаяСсылка());

ТЗСотрудникиДляСоздания = ТЗ.Скопировать(ОтборНужноСоздать);

## 18. Простой трюк для быстрого объединения таблиц значений

Стандартный способ объединения двух таблиц значений заключается в организации цикла, в котором строки второй таблицы приписываются в конец копии первой. Для этого используется следующий простой код:

ОбъединениеТаблиц = Таблица1.Скопировать();

Для Каждого Строка Из Таблица2 Цикл ЗаполнитьЗначенияСвойств(ОбъединениеТаблиц.Добавить(), Строка) КонецЦикла

## Работа со строками ТЗ

<https://forum.infostart.ru/forum9/topic236135/>

## Заполнить одну строку другой:

ЗаполнитьЗначенияСвойств(ТаблицаТоваров.Добавить(), ТаблицаТоваров[0])

***Пример:***

&НаКлиенте

Процедура ПартииВыбор(Элемент, ВыбраннаяСтрока, Поле, СтандартнаяОбработка)

СтандартнаяОбработка = Ложь;

ЗаполнитьЗначенияСвойств(Объект.Списание.Добавить(), Объект.Партии[ВыбраннаяСтрока]);

КонецПроцедуры

# ТабличныйДокумент

## Проверка на заполненность табличного документа

Если ТабличныйДокумент.Области.Количество()>0 Тогда

КонецЕсли;

# Описание типов

**Ссылка**: <https://koder.by/opisanie_tipov.php>

// Описание типа строки:  
НовСтрока = Новый ОписаниеТипов("Строка");  
// Описание строки с уточнением через квалификатор: максимальная длина строки = 50 символов  
КвалификаторыСтроки = Новый КвалификаторыСтроки(50);  
НовСтрока\_50 = Новый ОписаниеТипов("Строка", ,КвалификаторыСтроки);  
  
// Описание типа числа:  
НовЧисло = Новый ОписаниеТипов("Число");  
// Описание числа с уточнением через квалификатор: общее число разрядов = 14, число дробной части = 3  
КвалификаторыЧисла = Новый КвалификаторыЧисла(14, 3, ДопустимыйЗнак.Любой);  
НовЧисло\_14\_3 = Новый ОписаниеТипов("Число", КвалификаторыЧисла);  
  
// Описание типа даты:  
НовДата = Новый ОписаниеТипов("Дата");  
// Описание даты с уточнением через квалификатор: храниться только дата, без времени  
КвалификаторыДаты = Новый КвалификаторыДаты(ЧастиДаты.Дата);  
НовДата\_БезВремени = Новый ОписаниеТипов("Дата", , ,КвалификаторыДаты);  
  
// Описание типа булево:  
НовБулево = Новый ОписаниеТипов("Булево");// Истина, Ложь  
  
// Описание типа справочника:  
НовНоменклатура = Новый ОписаниеТипов("СправочникСсылка.Номенклатура");  
  
// Описание типа перечисление:  
НовСпособОплаты = Новый ОписаниеТипов("ПеречислениеСсылка.СпособыОплаты");  
  
// Описание типа документа:  
НовДоговор = Новый ОписаниеТипов("ДокументСсылка.Договор");  
  
// Описание типа структуры:  
НовСтруктура = Новый ОписаниеТипов("Структура");  
  
// Описание типа соответствия:  
НовСоответствие = Новый ОписаниеТипов("Соответствие");  
  
// Описание типа массива:  
НовМассив = Новый ОписаниеТипов("Массив");  
  
// Описание типа хранилище значения:  
НовХранилищеЗначения = Новый ОписаниеТипов("ХранилищеЗначения");  
  
// Описание типа таблица значений:  
НовТаблицаЗначений = Новый ОписаниеТипов("ТаблицаЗначений");  
  
// Описание типа список значений:  
НовТаблицаЗначений = Новый ОписаниеТипов("СписокЗначений");  
  
// Описание типа картинки:  
НовКартинка = Новый ОписаниеТипов("Картинка");  
  
// Описание типа уникального идентификатора:  
НовУникальныйИдентификатор = Новый ОписаниеТипов("УникальныйИдентификатор");  
  
// Описание типа объекта метаданых:  
НовОбъектМетаданных = Новый ОписаниеТипов("ОбъектМетаданных, Строка");  
  
// Описание составного типа (Например: строка+структура+справочник):  
СписокМассив = Новый Массив;  
СписокМассив.Добавить("Строка");  
СписокМассив.Добавить("Структура");  
СписокМассив.Добавить("СправочникСсылка.Номенклатура");  
ОписаниеСоставногоТипа = Новый ОписаниеТипов(СписокМассив);  
ЭтаФорма.РеквизитСписокЗначений.ТипЗначения = ОписаниеСоставногоТипа;

# Дерево в ТЗ

&НаСервере

Процедура ВТЗНаСервере()

тДерево = РеквизитФормыВЗначение("Действия");

тТаблица = РеквизитФормыВЗначение("ДействияТЗ");

ПреобразоватьВТЗРекурсия(тДерево, тТаблица, Новый УникальныйИдентификатор("00000000-0000-0000-0000-000000000000"));

тДерево.Строки.Очистить();

ЗначениеВРеквизитФормы(тТаблица, "ДействияТЗ");

ЗначениеВРеквизитФормы(тДерево, "Действия");

КонецПроцедуры

&НаСервере

Процедура ПреобразоватьВТЗРекурсия(тДерево, тТаблица, ГУИД)

Для Каждого тСтр Из тДерево.Строки Цикл

нСтр = тТаблица.Добавить();

нСтр.Представление = тСтр.Представление;

нСтр.Действие = тСтр.Действие;

нСтр.СрокПредставление = тСтр.СрокПредставление;

нСтр.ЭтоДействие = тСтр.ЭтоДействие;

нСтр.ЭтоУчастник = тСтр.ЭтоУчастник;

нСтр.НомерКартинки = тСтр.НомерКартинки;

нСтр.ПредставлениеРезультата = тСтр.ПредставлениеРезультата;

нСтр.ВидДействия = тСтр.ВидДействия;

нСтр.Состояние = тСтр.Состояние;

нСтр.ПредставлениеСотрудники = тСтр.ПредставлениеСотрудники;

нСтр.ПредставлениеЧистое = тСтр.ПредставлениеЧистое;

нСтр.ПометкаУдаления = тСтр.ПометкаУдаления;

нСтр.ДатаУстановкиСостояния = тСтр.ДатаУстановкиСостояния;

нСтр.Идентификатор = тСтр.Идентификатор;

нСтр.Недоступно = тСтр.Недоступно;

нСтр.НедоступноГруппа = тСтр.НедоступноГруппа;

нСтр.ЭтоПодзадача = тСтр.ЭтоПодзадача;

нСтр.ДействиеЗадачи = тСтр.ДействиеЗадачи;

нСтр.СостояниеКонтроля = тСтр.СостояниеКонтроля;

нСтр.РезультатТекстом = тСтр.РезультатТекстом;

нСтр.ЕстьФайлыОтчета = тСтр.ЕстьФайлыОтчета;

нСтр.ДействиеID = тСтр.ДействиеID;

нСтр.ДействиеТип = тСтр.ДействиеТип;

нСтр.ВидДействияID = тСтр.ВидДействияID;

нСтр.ВидДействияТип = тСтр.ВидДействияТип;

нСтр.ДействиеЗадачиID = тСтр.ДействиеЗадачиID;

нСтр.ДействиеЗадачи = тСтр.ДействиеЗадачи;

нСтр.НеУдалосьРазыменовать = тСтр.НеУдалосьРазыменовать;

нСтр.БудетУдален = тСтр.БудетУдален;

нСтр.Результат = тСтр.Результат;

нСтр.Родитель = ГУИД;

нСтр.ГУИД = Новый УникальныйИдентификатор();

//Если тСтр.Строки.Количество()>0 Тогда

// ПреобразоватьВТЗРекурсия(тСтр, тТаблица, нСтр.ГУИД);

//КонецЕсли;

КонецЦикла;

КонецПроцедуры

# ТЗ в Массив

Функция ТаблицаЗначенийВМассив(ТаблицаЗначений) Экспорт

Массив = Новый Массив();

СтруктураСтрокой = "";

НужнаЗапятая = Ложь;

Для Каждого Колонка Из ТаблицаЗначений.Колонки Цикл

Если НужнаЗапятая Тогда

СтруктураСтрокой = СтруктураСтрокой + ",";

КонецЕсли;

СтруктураСтрокой = СтруктураСтрокой + Колонка.Имя;

НужнаЗапятая = Истина;

КонецЦикла;

Для Каждого Строка Из ТаблицаЗначений Цикл

НоваяСтрока = Новый Структура(СтруктураСтрокой);

ЗаполнитьЗначенияСвойств(НоваяСтрока, Строка);

Массив.Добавить(НоваяСтрока);

КонецЦикла;

Возврат Массив;

КонецФункции

# Двоичные данные

## Из двоичных данных в ТабДок

&НаКлиенте

Процедура Команда1(Команда)

ТабДок = Новый ТабличныйДокумент;

ДвоичныеДанные = Новый ДвоичныеДанные("D:\Отчет.mxl");

Адрес = ПоместитьВоВременноеХранилище(ДвоичныеДанные);

ПрочитатьНаСервере(ТабДок, Адрес);

ТабДок.Показать();

КонецПроцедуры

&НаСервереБезКонтекста

Процедура ПрочитатьНаСервере(ТабДок, Адрес)

ДвоичныеДанные = ПолучитьИзВременногоХранилища(Адрес);

Поток = ДвоичныеДанные.ОткрытьПотокДляЧтения();

ТабДок.Прочитать(Поток);

Поток.Закрыть();

КонецПроцедуры

## Двоичные данные в ПДФ

ВремФайл = ПолучитьИмяВременногоФайла("pdf");

ТабличныйДокумент.Записать(ВремФайл, ТипФайлаТабличногоДокумента.PDF);

ДвоичныеЛистаСогласования = Новый ДвоичныеДанные(ВремФайл);

УдалитьФайлы(ВремФайл);

АдресДвоичныхДанныхЛистаСогласования = ПоместитьВоВременноеХранилище(ДвоичныеЛистаСогласования);

# СООБЩЕНИЯ

## Вопрос РежимДиалогаВопрос.ДаНет

Режим = РежимДиалогаВопрос.ДаНет;

Ответ = Вопрос(НСтр("ru = 'Вы уверены, что хотите создать электронный документ и запустить его обработку?'"), Режим, 0);

Если Ответ = КодВозвратаДиалога.Да Тогда

HTTPЗапрос();

Иначе

Возврат;

КонецЕсли;

## СообщениеПользователю

### БСП

// 1. Для вывода сообщения у поля управляемой формы, связанного с реквизитом объекта:  
ОбщегоНазначенияКлиент.СообщитьПользователю(  
НСтр("ru = 'Сообщение об ошибке.'"),,"ПолеВРеквизитеФормыОбъект", "Объект");  
// Альтернативный вариант использования в форме объекта:  
ОбщегоНазначенияКлиент.СообщитьПользователю(НСтр("ru = 'Сообщение об ошибке.'"), , "Объект.ПолеВРеквизитеФормыОбъект");  
  
// 2. Для вывода сообщения рядом с полем управляемой формы, связанным с реквизитом формы:  
ОбщегоНазначенияКлиент.СообщитьПользователю(НСтр("ru = 'Сообщение об ошибке.'"), ,"ИмяРеквизитаФормы"));  
  
// 3. Для вывода сообщения связанного с объектом информационной базы:  
ОбщегоНазначенияКлиент.СообщитьПользователю(НСтр("ru = 'Сообщение об ошибке.'"), ОбъектИнформационнойБазы, "Ответственный",,Отказ);  
  
// 4. Для вывода сообщения по ссылке на объект информационной базы:  
ОбщегоНазначенияКлиент.СообщитьПользователю(НСтр("ru = 'Сообщение об ошибке.'"), Ссылка, , , Отказ);  
  
// 5. Пример вызова на сервере:

ОбщегоНазначения.СообщитьПользователю(НСтр("ru = 'Объект с таким представлением уже существует.'",, Поле,, Отказ));

### СообщениеПользователю

Сообщение = Новый СообщениеПользователю;

Сообщение.Текст = “ ”;

Сообщение.ПутьКДанным = “Объект”;

Сообщение.КлючДанных = ЭтотОбъект.Ссылка;

Сообщение.Сообщить();