Универсальный интерфейс для управления фронт-офисными модулями «Restaurant Retail Control»

из любых конфигураций 1С7.7 или 1С8.х, в т. ч. из любых внешних приложений

(COM-объект RRC_back.dll)

v1.0.0.2

Назначение:

COM-объект обеспечивает для конфигураций 1С 7.7 и 1С 8.х, а так же для любого приложения, управление фронт-офисными модулями Minimarket (розничная торговля) и ТРАКТІРЪ (ресторан, кафе) программного комплекса Restaurant Retail Control.
COM-объект содержит все необходимые для бэк-офисного приложения методы управления справочниками, а так же чтение продаж на сервере фронт-офисного модуля.

Подключение к COM-объекту из Вашей конфигурации 1С7.7 или 1С8.х (или любого приложения):

COM-объект устанавливается в составе основного дистрибутива программы, и представляет из себя библиотеку RRC_back.dll , уже зарегистрированную в реестре. *(в противном случае можно зарегистрировать её из командной строки командой regsvr32 C:\Minimarket\RRC_back .dll)*

COM-объект содержит класс "RRCback", находящийся в пространстве имён "RRC_back". Соответственно, чтобы подключить его к Вашему приложению следует создать в приложении объект, например:

```
для  C++:        Front = new RRC_back.RRCback();
для  VB:         Set Front = CreateObject("RRC_back.RRCback")
для  1C7.7:      Front = СоздатьОбъект("RRC_back.RRCback");
для  1C8.x:      Front = Новый COMОбъект("RRC_back.RRCback");
```

Один из методов COM-объекта работает с объектом в виде набора записей ADODB.Recordset, поэтому, если Ваша учётная программа не является конфигурацией 1С 7.7 или 1С 8.х, то к Вашему приложению должен быть подключен так же пакет 'Microsoft ActiveX Data Objects Library'

При подключении (создании экземпляра COM-объекта) будет открыто соединение с сервером фронт-офисного модуля в соответствии с сохранёнными ранее настройками источника данных ODBC. На панели задач появится иконка, сигнализирующая о состоянии соединения.

Для корректного отключения от сервера необходимо просто разрушить объект (например : `delete Front; Set Front = Nothing  Front = 0;` или `Front = Null;` соответственно) - при разрушении объекта он сам закроет ODBC соединение с сервером фронт-офисного модуля.

Описание COM-объекта:

1. Товары (методы AddPluRest_, AddPluRetail_, AddPluFull_, DeletePlu_, DeleteAllPlu_)

AddPluRest_(cod, name, price, nds, kat, akz, loc)
для ресторана
метод добавляет запись про товар на сервер фронт-офиса ресторана, а если такая запись на сервере уже существует, то обновляет ее; ключевым полем для записи про товар является поле cod
метод **принимает** параметры:

cod (число, тип long от 1 до 999999) – код товара
name (строка, длина 50) – наименование товара
price (число, тип long) – розничная цена товара * 100 (в копейках)
nds (число, тип byte 0 - 6) – код ставки налога в фискальном регистраторе, если не подключен или без ндс, то nds=0
kat (число, тип byte) – для дисконтных карт: код категории скидки товарной группы, если не выделена, то kat=0
akz (число, тип byte) – при работе ресторана от двух СПД: для товара, по которому должен быть распечатан фискальный чек akz =1, для остальных товаров akz =0, если один СПД – akz =0
loc (число, тип byte) – признак локализации (места печати встречки): значение 0 = бар, значения 1, 2, 3 = соответств. номеру кухни, а значения 11 12 13 21 22 23 31 32 33 позволяют разбивать встречки на каждой кухне по трём цехам.

ответ — (тип boolean) True или False в зависимости от результата

AddPluRetail_(cod, cod1, bar, name, price, ost, nds, kat, akz, sig, ves)
для розничного магазина
метод добавляет запись про товар на сервер фронт-офиса, а если такая запись на сервере уже существует, то обновляет ее; ключевыми полями для записи про товар являются поля cod и cod1; количество записей про товар для каждого товара соответствует количеству штрихкодов этого товара, если штрихкода нет, то запись одна
метод **принимает** параметры:

cod (число, тип long от 1 до 999999) – код товара
cod1 (число, тип byte) – порядковый номер записи, если у товара нет штрихкода или всего один штрихкод: cod1=1, если штрихкодов у товара несколько, то у каждой записи свой cod1= порядковый номер штрихкода, принадлежащего товару.
bar (строка, длина 20) – штрихкод товара, если его нет — пустая строка
name (строка, длина 50) – наименование товара
price (число, тип long) – розничная цена товара * 100 (в копейках)
ost (число, тип long) – текущий остаток товара * 1000
nds (число, тип byte 0-6) – код ставки налога в фискальном регистраторе, если не подключен или безндс, то nds=0
kat (число, тип byte) – для дисконтных карт: код категории скидки товарной группы, если не выделена, то kat=0
akz (число, тип byte) – при работе магазина от двух СПД: для товара, по которому должен быть распечатан фискальный чек akz =1, для остальных товаров akz =0, если один СПД – akz=0
sig (число, тип byte) – для продажи сигарет по цене указанной на пачке: если товар - сигареты, то sig=1, иначе sig=0
ves (число, тип byte) – признак весового товара: штучный ves=0, весовой ves=1 (только для работы с торговыми весами, подключенными к рабочему месту кассира)

ответ — (тип boolean) True или False в зависимости от результата

AddPluFull_(cod, cod1, bar, name, price, ost, nds, kat, akz, sig, loc, opt, tara, ves, dat)
метод предназначен для аптек, оптово-розничных магазинов, магазинов где чекопечатающие весы будут загружаться из фронт-офисного модуля и торговых точек, где используется калькуляция производства (например кофемашина, мясное производство, разливное пиво)
метод объединяет два предыдущих: добавляет запись про товар на сервер фронт-офиса, а если такая запись на сервере уже существует, то обновляет ее
метод, кроме описанных выше, **принимает** дополнительные параметры:
opt (число, тип long) – если нужна вторая цена на кассе для оптово-розничного магазина: оптовая цена товара * 1000 (в десятых частях копейки), если не используется, то opt=0
tara (число, тип long) – признак «дробного» товара для аптеки: обычный товар tara=0, «дробный» товар tara= количеству блистеров в коробке * 1000
ves (число, тип long) – для загрузки чекопечатающих весов из фронт-офисного модуля магазина ves=Код На Весах, для работы с торговыми весами ves=1, если товар штучный - ves=0
dat (строка, длина 10) – только для загрузки чекопечатающих весов из фронт-офисного модуля магазина: срок годности для этикетки весов — например 31.12.2019, если товар штучный - пустая строка.
ответ — (тип boolean) True или False в зависимости от результата

DeletePlu_(cod)
метод удаляет все записи про товар с выбранным кодом на сервере фронт-офиса
метод **принимает** параметр: **cod** (число, тип long от 1 до 999999) – код товара, который удаляем
ответ — (тип boolean) True или False в зависимости от результата

DeleteAllPlu_( )
метод удаляет все записи про ВСЕ товары на сервере фронт-офиса. (можно вместо него использовать метод DeletePlu_(0))
ответ — (тип boolean) True или False в зависимости от результата

РЕКОМЕНДАЦИИ:
В наших конфигурациях для 1С7.7 и 1С8.х в справочнике Товары есть кнопки:
ДобавитьВсеТовары для перезаписи всего справочника товаров: сначала удаляем все товары DeleteAllPlu_, затем через цикл добавляем по-одному с помощью одного из методов AddPluRest_, AddPluRetail_, AddPluFull_
ДобавитьТовар для добавления (обновления) текущего товара с помощью методов AddPluRest_, AddPluRetail_, AddPluFull_
УдалитьТовар для удаления текущего товара с помощью DeletePlu_
А также в каждом документе есть процедура **ОбновитьНаКассах** для обновления табличной части документа после его проведения через цикл построчно с помощью одного из методов AddPluRest_, AddPluRetail_, AddPluFull_ (см. пример)

2. Продажи (методы GetSale_ UpdateSale_)

GetSale_(dat, smn)
предназначен для получения продаж из сервера фронт-офиса в учётную программу
метод загружает продажи за выбранную дату из таблицы продаж в объект ADODB.Recordset
Метод **принимает** параметры:
dat (строка, длина 10) – дата продаж в формате Год/Месяц/День, например 2019/01/01
smn (число, тип byte) – начало смены, соответствует началу часа, т. е. число от 0 до 23 — момент суток с которого будут считаны продажи (например, если smn=8, то продажи будут прочитаны с 08:00:00 по 07:59:59 следующих суток)
ответ — продажи за выбранную дату: набор записей ADODB.Recordset, который содержит поля:
field(0) – **cod** число – код товара
field(1) – **name** строка – наименование товара
field(2) – **kol** число – количество * 1000
field(3) – **price** число – цена продажи * 1000
field(4) – **skid** число – сумма скидки * 1000, если нет — skid=0
field(5) – **card** число – код (порядковый номер) дисконтной карты, если нет — card=0
field(6) – **fiscal** число – флаг того, что товар был отправлен на фискальный регистратор (1-да, NULL-нет)
в случае ошибки метод ничего не возвращает

UpdateSale_(dat, smn, msg)
только для розничного магазина
метод обнуляет текущие продажи на сервере фронт-офиса магазина после закрытия продаж за смену (списания товаров в реализацию)
метод **принимает** параметры:
dat (строка, длина 10) – дата продаж в формате Год/Месяц/День, например 2019/01/01
smn (число, тип byte) – начало смены, соответствует началу часа, т. е. число от 0 до 23 — момент суток с которого будут считаны продажи (например, если smn=8, то продажи будут прочитаны с 08:00:00 по 07:59:59 следующих суток)
msg (число, тип byte) – флаг вывода сообщения по окончании обработки на сервере (0-нет, 1-да)
ответ — (тип boolean) True или False в зависимости от результата

РЕКОМЕНДАЦИИ:
В наших конфигурациях обработка Закрытие Продаж получает продажи методом GetSale_ и создаёт документ реализации, после проведения которого необходимо не только обновить его табличную часть через цикл с помощью одного из методов AddPluRest_, AddPluRetail_, AddPluFull_ (см. пример)

Для розничных магазинов, где в фронт-офисе выводятся актуальные остатки товаров, следует так же использовать метод UpdateSale_ (который обнуляет текущие продажи на фронт-офисе с учётом изменившихся остатков после проведения документа реализации)

Если у Вас накопительная или бонусная дисконтная система скидок, то Вам так же следует переписать Дисконтные карты методами DelAllCard_ и AddCard_ , чтобы обороты по картам попали на сервер фронт-офиса

3. Дисконтные карты (методы AddCard_ AddTypeCard_ DelAllCard_ DelAllTypeCard_)

AddCard_(card, bar, typ, total)
метод добавляет на сервер фронт-офиса запись про одну Дисконтную карту
Метод **принимает** параметры:
card (число, тип long) – код (порядковый номер) дисконтной карты в учётной системе
bar (строка, длина 20) – штрихкод дисконтной карты
typ (число, тип long) – код типа карты: для бонусных карт = 0, для карт с фиксированной скидкой от 1 до 100 (% скидки на карте), для накопительных карт число от 101 до 999 (тип карты)
total (число, тип long) – сумма накопления на карте * 100 (в копейках) — только для накопительных карт, иначе 0
ответ — (тип boolean) True или False в зависимости от результата

DelAllCard_( )
метод удаляет все записи про ВСЕ Дисконтные карты на сервере фронт-офиса.
ответ — (тип boolean) True или False в зависимости от результата

AddTypeCard_(id, typ, kat, skid, total)
применяется только для накопительной дисконтной системы
метод добавляет на сервер фронт-офиса запись про одно Условие из справочника Типы Дисконтных карт (набор условий в разрезе категорий товарных групп и в разрезе накоплений на картах, по которым должны работать дисконтные карты)
Метод **принимает** параметр:
id (число, тип byte) – код (порядковый номер) условия
typ (число, тип long, в диапазоне: от 101 до 999) – код типа дисконтной карты
kat (число, тип byte) – код категории скидки товарной группы, если категория не выделена (все остальные товары), то kat=0
skid (число, тип byte) – размер скидки для данного условия в процентах (целое число)
total (число, тип long) – сумма накопления на карте * 100 (в копейках), при которой начинает действовать это условие
ответ — (тип long) количество загруженных записей

DelAllTypeCard_( )
метод удаляет все записи про ВСЕ условия для накопительных дисконтных карт на сервере фронт-офиса.
ответ — (тип boolean) True или False в зависимости от результата

РЕКОМЕНДАЦИИ:
Накопительная дисконтная система, (скидки в разрезе товарных категорий и накопления на карте) использует на сервере фронт-офиса две таблицы Дисконтные карты и Типы дисконтных карт (условия), поэтому необходимо их загружать отдельно. Алгоритм загрузки:
1. методами DelAllCard_ и DelAllTypeCard_ очищаем таблицы дисконтных карт и таблицу типов карт
2. через цикл записываем данные в таблицы методами AddCard_ и AddTypeCard_
В дальнейшем достаточно перезаписывать только таблицу дисконтных карт (DelAllCard_ и AddCard_)

Если Вы не планируете использовать Дисконтные карты с накопительной системой скидок и скидками в разрезе товарных категорий, и Вам нужны просто **дисконтные карты с фиксированным процентом на весь ассортимент**, то можно выгрузить только список Дисконтных карт методами DelAllCard_ и AddCard_ , где в параметр **typ** вместо Типа карты передать размер скидки в % (число от 1 до 100).
В этом случае фронт-офис будет обрабатывать дисконтную карту как фиксированную скидку на весь ассортимент. Если в параметр **typ** передать 0, то карта будет работать как **бонусная дисконтная карта** (просто накапливает бонусы, которыми впоследствии можно будет оплатить часть чека)

ПРИМЕР построения **накопительной дисконтной системы** скидок в разрезе товарных категорий**:**
Построим накопительную дисконтную схему для двух типов дисконтных карт:
«для пионеров» - код типа карты type=101, и «для пенсионеров» type=102
Так же из общей товарной массы выделим три категории скидок товаров:
«алкоголь» - код категории скидки kat=1, «конфеты» kat=2, «хлеб» kat=3, соответственно все остальные товары, которые не принадлежат ни к какой категории имеют kat=0
Задача:
на «алкоголь» скидок нет вообще, на все остальные товары фиксированная скидка 2%,
на «конфеты» скидка есть только у «пионеров» накопительная — сначала 3%, а от 1000 на карте уже 5%,
на «хлеб» у «пионеров» фиксированная 1%, а у «пенсионеров» - сначала 1%, от 2000 на карте уже 2%, а от 5000 на карте 5%

справочник Типы Дисконтных карт будет выглядеть так:

	id	type	kat	skid	total
тип 101 (пионер) на все остальные товары kat=0 скидка 2% от суммы на карте = 0.00	1	101	0	2	0
тип 101 (пионер) на хлеб kat=3 скидка 1% от суммы на карте = 0.00	2	101	3	1	0
тип 101 (пионер) на конфеты kat=2 скидка 3% от суммы на карте = 0.00 (1-я ступень)	3	101	2	3	0
тип 101 (пионер) на конфеты kat=2 скидка 5% от суммы на карте = 1000.00 (2-я ступень)	4	101	2	5	100000
тип 102 (пенсионер) на все остальные товары kat=0 скидка 2% от суммы на карте = 0.00	5	102	0	2	0
тип 102 (пенсионер) на хлеб kat=3 скидка 1% от суммы на карте = 0.00 (1-я ступень)	6	102	3	1	0
тип 102 (пенсионер) на хлеб kat=3 скидка 2% от суммы на карте = 2000.00 (2-я ступень)	7	102	3	2	200000
тип 102 (пенсионер) на хлеб kat=3 скидка 5% от суммы на карте = 5000.00 (3-я ступень)	8	102	3	5	500000

«алкоголь» kat=1 отсутствует везде, а условия для «конфет» kat=2 есть только у типа карты 101 (пионер)

Примеры использования методов:

1С7.7: процедура обновления табличной части документа (метод AddPluRetail_ , на товаре несколько штрихкодов)

```
Процедура ОбновитьНаКассах(Контекст)  Экспорт
	Контекст.ВыбратьСтроки();
	Пока Контекст.ПолучитьСтроку() = 1 Цикл
		cod = Контекст.Товар.ВнутрКод;
		name = Контекст.Товар.Наименование;
		price = глПолучитьЦену(Контекст.Товар,Константа.РозничныйСклад.КатегорияЦен,ТекущаяДата())*100;
		ost = Рег.СводныйОстаток(Константа.РозничныйСклад,Контекст.Товар,,"Количество")*1000;
		nds = Контекст.Товар.СтавкаНДС.Код;
		kat = Контекст.Товар.КатегорияСкидки.Код;
		akz = Контекст.Товар.Акцизный;
		sig = Контекст.Товар.Сигареты;
		ves = Контекст.Товар.Весовой;
		ШК = СоздатьОбъект("Справочник.ШтрихКоды");
		ШК.ИспользоватьВладельца(Контекст.Товар);
		ШК.ВыбратьЭлементы();
		ЧислоШтрихКодов = 0;
		Пока ШК.ПолучитьЭлемент() = 1 Цикл  // перебираем штрихкоды из подчинённого справочника
			ЧислоШтрихКодов = ЧислоШтрихКодов + 1;
			Front.AddPluRetail_(cod,ЧислоШтрихКодов,ШК.ШтрихКод,name,price,ost,nds,kat,akz,sig,ves);
		КонецЦикла;
		Если ЧислоШтрихКодов = 0 Тогда      // если штрихкодов нет, создаём строку с пустым штрихкодом
			Front.AddPluRetail_(cod,1,"",name,price,ost,nds,kat,akz,sig,ves);
		КонецЕсли;
	КонецЦикла;
КонецПроцедуры
```

1С8.х: функция получает продажи из объекта ADODB.RecordSet в объект ТаблицаЗначений (метод GetSale_)

```
Функция ПолучитьПродажи(ВыбДата,НачСмены,ТабЗнач) Экспорт
	Перем RecordSet, Front;
	ТабЗнач.Очистить();
	ТабЗнач.Колонки.Очистить();
	ДатаВыборки = Формат(ВыбДата, "ДФ=ггггММдд");
	День = Прав(ДатаВыборки,2);
	Месяц = Сред(ДатаВыборки,5,2);
	Год = Лев(ДатаВыборки,4);
	dat = Год + "/" + Месяц + "/" + День;
	smn = Число(НачСмены);
	RecordSet = Новый COMОбъект("ADODB.RecordSet");          // создаём новый RecordSet и COM-объект
	Front = Новый COMОбъект("RRC_back.RRCback");
	RecordSet = Front.GetSale_(dat, smn);                    // получаем в RecordSet продажи с сервера
	Если RecordSet.RecordCount() = 0 Тогда                   // проверяем что RecordSet не пустой
		Предупреждение("Нет продаж !!!");
		Возврат Ложь;
	КонецЕсли;
  // в нашей таблице создаём новые колонки по числу колонок RecordSet и называем их так же, как поля RecordSet
	Для iCount = 0 По RecordSet.Fields.Count-1 Цикл
		ТабЗнач.Колонки.Добавить(RecordSet.Fields(iCount).Name);
	КонецЦикла;
  // обращаться к колонкам таблицы теперь можно по имени поля RecordSet, например СтрТЗ["price"], СтрТЗ["cod"]
	RecordSet.MoveFirst();
	Пока RecordSet.EOF() = 0 Цикл                             // заполняем таблицу значениями из RecordSet
		СтрТЗ = ТабЗнач.Добавить();
		Для iCount = 0 По RecordSet.Fields.Count-1 Цикл
			СтрТЗ[RecordSet.Fields(iCount).Name] = RecordSet.Fields(iCount).Value;
		КонецЦикла;
	RecordSet.MoveNext();
	КонецЦикла;
	RecordSet.Close();                                        // закрываем RecordSet
RecordSet = Null;                                             // уничтожаем объекты
Front = Null;
Возврат Истина;
КонецФункции
```