**Структура программы «ContractsRegistration» для Бизнес Связь**

1. Инициализация программы.

Создаем датамодуль, затем, если из этого модуля успешно связываемся с Оракл, запускаем главную форму.

Application.CreateForm(TDm, Dm);

if Dm.OraSession.Connected then

begin

if Dm.modal\_rez\_ChangePassword <> 2 then

// idCancel = 2 смена пароля неудачна, выходим

begin

Application.Initialize;

Application.Title := 'Учет абонентов сотовой связи';

Application.CreateForm(TMainForm, MainForm);

Application.Run;

end;

end

else

begin

Dm.Free;

end;

После создания новой формы, из этого кода – обязательно убирать.

1. Главная форма.

Вызов почти всех событий в программе реализован через ActionList .

Хотелось иметь механизм, позволяющий унифицировать вызов новых форм и не плодить одни и те же команды.

Для этого сделано следующее –

* Создан список классов.

type

TMyRef = class of TChildForm;

Type

TListClass = array of TMyRef;

procedure TMainForm.FormCreate(Sender: TObject);

begin

ListClass := TListClass.Create(TRefUserNamesForm, TRefMobilOperatorForm, TRefAccountsForm, TRefPhonesForm, TRefPhonesOnAccountForm,

// 1 2 3 4 5

* Каждый класс имеет в списке свой номер
* Каждый action имеет tag c номером, совпадающим со списком классов.
* Каждый action вызывает одну и ту же процедуру.

procedure TMainForm.CommonClickMenu(Sender: TObject);

var

act: TAction;

begin

act := Sender AS TAction;

itm := act.Tag;

NameForm := act.Caption;

fr := TMyRef(ListClass[(itm - 1)]);

SetActiveScreen;

Resizing(wsNormal);

end;

и

MDIChildFound := False;

for i := 0 to MDIChildCount - 1 do

if MDIChildren[i] is fr then

begin

MDIChildren[i].Show;

MDIChildFound := True;

Break;

end;

if (not MDIChildFound) and (itm <> 0) then

begin

New\_F := fr.Create(ChildForm,itm, New\_F, self.Handle, NameForm);

Dm.CurrentWnd := New\_F.Handle;

New\_F.Show;

end;

1. Вызов форм осуществляется в двух состояниях формы – или модальном, или как МДИ дочерняя.

Реализация – Любая форма создается как наследник от класса TChildForm

3 конструктора

constructor Create(AOwner: TComponent); overload; override;

constructor Create(AOwner: TComponent; var AFormVar: TChildForm; Cpt: string; ShM: Boolean); reintroduce; overload;

constructor Create(AOwner: TComponent; Itm: Integer; var AFormVar: TChildForm; dscr: HWND; Cpt: string); reintroduce; overload;

позволяют создать разные формы

кроме этого создается режим формы –

property FormMode: TFormMode read GetFormMode write SetFormMode;

Type

TFormMode = (fmInactive, fmBrowse, fmInsert, fmEdit);

property OnSetFormMode: TNotifyEvent read FOnSetFormMode write FOnSetFormMode;

позволяет реагировать на событие

1. Все общие компоненты, относящиеся к работе всей программы, находятся в датамодуле uDm.

Здесь же датасеты от справочников для повторного применения кода. Так мы ссылаясь в форме на датасет справочника, сразу видим изменение данных без повторного обращения к базе.

Ссылки на функции и типы находятся в модуле Func\_Const. Ссылки на создание экселовских и пдф в юните ExportGridToExcelPDF.

1. Все сделанные справочники имеют префикс Ref и наследованы от класса TRefForm = class(TChildForm).

Создание справочников

* . – наследуемся от TRefForm. Датасет отсправочника располагаем в датамодуле.
* Создаем поля в FieldsEditor. Если поле не надо показывать – делаем его невидимым. Прописываем в DisplayLabel название поля – это будет названием столбца в таблице справочника. Можно использовать lookup поля. Если lookup поле типа Boolean – будет отображен чекбокс, если поле типа string – то будет отображен комбобокс.
* Если создали поля USER\_CREATED\_FIO, DATE\_CREATED\_, USER\_LAST\_UPDATED\_FIO, DATE\_LAST\_UPDATED\_ - именно с такими названиями, - то эти поля будут показаны в зависимости от настроек системы.
* Если справочник простой, то просто назначаем датасет формы датасету в датамодуле. Это делается до inherited; Затем, при необходимости, назначаем события. Например: qRef.AfterInsert := qRefAfterInsert; qRef – это имя датасета формы.
* Если справочник сложный (имеет много lookup полей из других справочников), то дополнительно назначаем список классов аналогично главной форме ListCls := TLCls.Create(TRefVirtual\_OperatorsForm, TRefPhonesOnAccountForm, TRefFilialForm, TRefAbonentsForm); и список заголовков форм

inherited;

CaptList.Add('Справочник лицевых счетов виртуального оператора');

CaptList.Add('Справочник телефонов на лицевых счетах');

//CaptList.Add('Справочник тарифов');

CaptList.Add('Справочник филиалов');

CaptList.Add('Справочник абонентов');

Порядок списков должен соответствовать списку полей в датасете в датамодуле.

Каждый класс – соответствует вызываемой форме при редактировании данных

* В шаблоне справочника реализован метод доставки сообщений.

procedure MessageReceiverMoveNext(var Msg: TMessage); message WM\_NOTIFY\_GO\_ON\_THE;

procedure TRefForm.MessageReceiverMoveNext(var Msg: TMessage);

//var

//txt: PChar;

begin

//txt := PChar(Msg.lParam);

Msg.Result := 0;

aMoveNext.Caption := Dm.MoveNextHint;

aMoveNext.Hint := aMoveNext.Caption;

btnMoveNext.Hint := aMoveNext.Caption;

aMovePrev.Caption := Dm.MovePrevHint;

aMovePrev.Hint := aMovePrev.Caption;

btnMovePrev.Hint := aMovePrev.Caption;

* Назначение датасета и списка подключаемых классов до наследования обуславливается свойствами

property qRef: TOraQuery read fqRef write SetqRef;

property dsqRef: TOraDataSource read fdsqRef write SetdsqRef;

property ListCls: TLCls read fListCls write SetListCls;

property CaptList: TStringList read fCaptList write fCaptList;

property EditingData : Boolean read fEditingData write SetEditingData;

1. Все журналы имеют префикс Rep и наследованы от класса TRepFrm = class(TChildForm)

Имеют свойства

property cb\_Mob\_Oper : Boolean read fcb\_Mob\_Oper write Setcb\_Mob\_Oper;

property cb\_Period : Boolean read fcb\_Period write Setcb\_Period;

property CLBAccounts : Boolean read fCLBAccounts write SetCLBAccounts;

property CLB\_VirtAccs : Boolean read fCLB\_VirtAccs write SetCLB\_VirtAccs;

property CLB\_FilialT : Boolean read fCLB\_filial write SetCLB\_filial;

property b\_PhoneField : Boolean read fb\_PhoneField write Setb\_PhoneField;

property panels\_visible\_all : Boolean read fpanels\_visible\_all write set\_panels\_visible\_all;

эти свойства отвечают за видимость каждого фильтра

журналы имеют свой собственный датасет qReport.

1. База данных

***SCHEDULER JOBS***

Суть – юридические лица

J\_GET\_VIRT\_ACCOUNT\_FROM\_FTP - GET\_VIRT\_ACCOUNT\_FROM\_FTP;

Загрузка файлов виртуальных счетов с FTP

J\_GET\_DBF\_VIRT\_ACCOUNT - virt\_account\_temp\_load\_dbf;

Загрузка данных из DBF файлов во временную таблицу VIRT\_ACCOUNT\_TEMP (первый парсер данных)

J\_PARSE\_VIRT\_ACCOUNT - PARSE\_VIRT\_ACCOUNT\_FILES;

Разбор данных из дбф файла с загрузкой в VIRTUAL\_ACCOUNTS, PHONES и PHONE\_ON\_ACCOUNTS из временной таблицы VIRT\_ACCOUNT\_TEMP

платежи

J\_GET\_PAYMENTS\_FROM\_FTP - GET\_PAYMENTS\_FROM\_FTP;

Загрузка файлов платежей с FTP

J\_LOAD\_PAYMENTS\_FILES - LOAD\_PAYMENTS\_FILES;

Загрузка данных из файлов платежей

счета

J\_PARSE\_BILL\_FILES - P\_PARSE\_BILL\_FILES;

Отправка новых файлов счетов в парсер

J\_PARSE\_BILLS - P\_PARSE\_BILLS;

Разбор файлов после парсера

J\_SEND\_DBF\_TO\_FTP - SEND\_DBF\_TO\_FTP;

Отправляет на FTP в папку 1C/TO\_1C

файлы с расширением .dbf из D:\Tarifer\DB\FOR\_1C.

Отправленные файлы сохраняет в папке D:\Tarifer\DB\FOR\_1C\BACKUP\'