**Приложение Б**

**Текст программы**

Текст программы LinkUp главный модуль

unit ULinkd;

interface

uses

Winapi.Windows, Winapi.Messages, System.SysUtils, System.Variants,

System.Classes, Vcl.Graphics,

Vcl.Controls, Vcl.Forms, Vcl.Dialogs, ShellApi, Vcl.Menus, Vcl.StdCtrls,

Vcl.ExtCtrls, IniFiles, IdBaseComponent, IdComponent, IdTCPConnection,

IdTCPClient, IdExplicitTLSClientServerBase, IdFTP, IOUtils, FileCtrl,

DateUtils, ustr;

type

TFMain = class(TForm)

PopupMenu1: TPopupMenu;

N1: TMenuItem;

N2: TMenuItem;

GroupBox1: TGroupBox;

GroupBox2: TGroupBox;

RadioButton1: TRadioButton;

RadioButton2: TRadioButton;

Edit1: TEdit;

Edit2: TEdit;

Edit3: TEdit;

BSelectLocalFolder: TButton;

BSelectUNCFolder: TButton;

BSettingFTP: TButton;

GroupBox3: TGroupBox;

BSaveSettings: TButton;

Label1: TLabel;

Edit4: TEdit;

CheckBox1: TCheckBox;

Timer1: TTimer;

IdFTP1: TIdFTP;

Label2: TLabel;

Edit5: TEdit;

BSelectArcFolder: TButton;

Label3: TLabel;

Label4: TLabel;

Edit6: TEdit;

Edit7: TEdit;

procedure FormCreate(Sender: TObject);

procedure N1Click(Sender: TObject);

procedure FormClose(Sender: TObject; var Action: TCloseAction);

procedure N2Click(Sender: TObject);

procedure BSaveSettingsClick(Sender: TObject);

procedure BSelectLocalFolderClick(Sender: TObject);

procedure BSelectUNCFolderClick(Sender: TObject);

procedure BSelectArcFolderClick(Sender: TObject);

procedure BSettingFTPClick(Sender: TObject);

procedure CheckBox1Click(Sender: TObject);

procedure Timer1Timer(Sender: TObject);

private

Procedure OnMinimizeProc(Sender: TObject);

public

Procedure LoadIni;

Procedure Ic(n: Integer; Icon: TIcon);

Procedure Log(f: string);

Procedure UploadFTPFiles(d: string; f: string);

Procedure UploadUNCFiles(d: string; f: string);

Procedure Status(f: string);

protected

Procedure ControlWindow(Var Msg: TMessage); message WM\_SYSCOMMAND;

Procedure IconMouse(var Msg: TMessage); message WM\_USER + 1;

end;

var

FMain: TFMain;

remotedir: string;

errorcount: Integer;

arcpath: string;

implementation

{$R \*.dfm}

uses Uftpsett;

Procedure TFMain.Log(f: string);

begin

// логируем

Label4.Caption:=f;

end;

Procedure TFMain.Status(f: string);

begin

Label4.Caption:=f;

end;

procedure TFMain.Timer1Timer(Sender: TObject);

begin

Timer1.Enabled:=false;

Log('Запуск обработки новых файлов');

if RadioButton2.Checked then UploadFTPFiles(Edit1.Text, '');

if RadioButton1.Checked then UploadUNCFiles(Edit1.Text, '');

Log('Ожидаю следующего запуска');

Timer1.Enabled:=true;

end;

Procedure TFMain.UploadUNCFiles(d: string; f: string);

var

FileName: string;

t:longbool;

Begin

try

Log('Копируем в UNC...');

for FileName in TDirectory.GetFiles(Edit1.text, '\*.\*') do

begin

Log('Закачиваем '+FileName);

edit6.text:= FileName;

edit7.text:= edit2.text+'\' + extractfilename(FileName);

t:=CopyFile(PWideChar(FileName), PWideChar(edit2.text+'\' + extractfilename(FileName)),false);

if t<>true then

Log('Ошибка копирования файла');

Log('Переносим в архив '+FileName);

deletefile(arcpath + extractfilename(FileName));

if Not MoveFile(PWideChar(FileName), PWideChar(arcpath + extractfilename(FileName))) then

Log('Ошибка переноса файла в архив');

end;

except

on E: exception do

begin

Log('Ошибка загрузки файлов');

E := Nil;

end;

end;

IdFTP1.DisConnect;

end;

Procedure TFMain.UploadFTPFiles(d: string; f: string);

var

FileName: string;

Begin

try

Log('Соединяюсь с сервером...');

IdFTP1 := TIdFTP.Create(nil);

IdFTP1.Host := fftpsett.edit1.text;

IdFTP1.Username := fftpsett.edit2.text;

IdFTP1.Password := fftpsett.edit3.text;

idFTP1.Passive:= True;

IdFTP1.Connect;

IdFTP1.ChangeDir('/'+edit3.text);

for FileName in TDirectory.GetFiles(Edit1.text, '\*.\*') do

begin

Log('Закачиваем '+FileName);

IdFTP1.Put( FileName);

Log('Переносим в архив '+FileName);

if Not MoveFile(PWideChar(FileName), PWideChar(arcpath + extractfilename(FileName))) then

Log('Ошибка переноса файла в архив');

end;

except

on E: exception do

begin

Log('Ошибка загрузки файлов');

E := Nil;

end;

end;

IdFTP1.DisConnect;

end;

Procedure TFMain.OnMinimizeProc(Sender: TObject);

Begin

PostMessage(Handle, WM\_SYSCOMMAND, SC\_MINIMIZE, 0);

End;

procedure TFMain.IconMouse(var Msg: TMessage);

Var

p: tpoint;

begin

GetCursorPos(p); // Запоминаем координаты курсора мыши

Case Msg.LParam OF // Проверяем какая кнопка была нажата

WM\_LBUTTONUP, WM\_LBUTTONDBLCLK

: { Действия, выполняемый по одинарному или двойному щелчку левой кнопки мыши на значке. В нашем случае это просто активация приложения }

Begin

Ic(2, Application.Icon); // Удаляем значок из трея

ShowWindow(Application.Handle, SW\_SHOW);

// Восстанавливаем кнопку программы

ShowWindow(Handle, SW\_SHOW); // Восстанавливаем окно программы

End;

WM\_RBUTTONUP

: { Действия, выполняемый по одинарному щелчку правой кнопки мыши }

Begin

SetForegroundWindow(Handle);

// Восстанавливаем программу в качестве переднего окна

PopupMenu1.Popup(p.X, p.Y);

// Заставляем всплыть тот самый TPopUp о котором я говорил чуть раньше

PostMessage(Handle, WM\_NULL, 0, 0);

end;

End;

end;

procedure TFMain.N1Click(Sender: TObject);

begin

Begin

Ic(2, Application.Icon); // Удаляем значок из трея

ShowWindow(Application.Handle, SW\_SHOW);

// Восстанавливаем кнопку программы

ShowWindow(Handle, SW\_SHOW); // Восстанавливаем окно программы

End;

end;

procedure TFMain.N2Click(Sender: TObject);

begin

Application.Terminate;

end;

Procedure TFMain.LoadIni;

var

ini: tinifile;

begin

ini := tinifile.Create(Extractfilepath(Application.ExeName) + '\' +

'linkup.ini');

try

Timer1.Interval := ini.ReadInteger('INTERVAL', 'mSec', 20000);

except

on E: exception do

begin

Timer1.Interval := 20000;

ShowMessage('В настройках нужно указать время таймера.');

E := Nil;

end;

end;

Edit1.Text := ini.ReadString('LOCAL', 'Path', '');

Edit2.Text := ini.ReadString('REMOTE', 'UNC', '');

Edit3.Text := ini.ReadString('REMOTE', 'FTPfolder', '');

Edit5.Text := ini.ReadString('LOCAL', 'Archive', '');

RadioButton1.checked := ini.ReadBool('REMOTE', 'TYPE\_UNC', True);

RadioButton2.checked :=Not RadioButton1.checked;

CheckBox1.checked := ini.ReadBool('SYNC', 'Enabled', True);

Timer1.Enabled:= CheckBox1.checked;

Edit4.Text := Inttostr(Round(Timer1.Interval / 1000));

ini.free;

remotedir := '';

end;

procedure TFMain.BSelectLocalFolderClick(Sender: TObject);

var

SelectedFolder: string;

begin

SelectDirectory('Выберите директорию', '', SelectedFolder);

Edit1.Text := SelectedFolder;

end;

procedure TFMain.BSelectUNCFolderClick(Sender: TObject);

var

SelectedFolder: string;

begin

SelectDirectory('Выберите директорию', '', SelectedFolder);

Edit2.Text := SelectedFolder;

end;

procedure TFMain.BSettingFTPClick(Sender: TObject);

begin

FFtpSett.ShowModal;

end;

procedure TFMain.BSaveSettingsClick(Sender: TObject);

var

ini: tinifile;

begin

ini := tinifile.Create(Extractfilepath(Application.ExeName) + '\' +

'linkup.ini');

try

Timer1.Interval := Strtoint(Edit4.Text) \* 1000;

Except

on E: exception do

begin

ini.free;

ShowMessage

('В настройках таймера допускается только целое число секунд.');

E := Nil;

Exit;

end;

end;

ini.WriteInteger('INTERVAL', 'mSec', Timer1.Interval);

ini.WriteString('LOCAL', 'Path', Edit1.Text);

ini.WriteString('LOCAL', 'Archive', Edit5.Text);

ini.WriteString('REMOTE', 'UNC', Edit2.Text);

ini.WriteString('REMOTE', 'FTPfolder', Edit3.Text);

ini.WriteBool('REMOTE', 'TYPE\_UNC', RadioButton1.checked);

ini.WriteBool('SYNC', 'Enabled', CheckBox1.checked);

ini.free;

end;

procedure TFMain.BSelectArcFolderClick(Sender: TObject);

var

SelectedFolder: string;

begin

SelectDirectory('Выберите директорию', '', SelectedFolder);

Edit5.Text := SelectedFolder;

end;

procedure TFMain.CheckBox1Click(Sender: TObject);

begin

Timer1.Enabled:=CheckBox1.Checked;

end;

Procedure TFMain.ControlWindow(Var Msg: TMessage);

Begin

IF Msg.WParam = SC\_MINIMIZE then

Begin

Ic(1, Application.Icon); // Добавляем значок в трей

ShowWindow(Handle, SW\_HIDE); // Скрываем программу

ShowWindow(Application.Handle, SW\_HIDE); // Скрываем кнопку с TaskBar'а

End

else

inherited;

End;

procedure TFMain.FormClose(Sender: TObject; var Action: TCloseAction);

begin

if MessageDlg('Вы действительно ходите закрыть?', mtInformation, [mbYes, mbNo] ,0) = mrYes

then Application.Terminate;

// а то задолбает вусмерть

Action := caNone;

Application.Minimize;

end;

procedure TFMain.FormCreate(Sender: TObject);

begin

Application.onMinimize := OnMinimizeProc;

LoadIni;

errorcount := 0;

arcpath := Edit5.Text + '\' + Inttostr(YearOf(date)) + '\' +

Inttostr(MonthOfTheYear(date)) + '\' + Inttostr(DayOfTheMonth(date)) + '\';

CreateDirEx(arcPath);

end;

Procedure TFMain.Ic(n: Integer; Icon: TIcon);

Var

Nim: TNotifyIconData;

begin

With Nim do

Begin

cbSize := Nim.SizeOf();

Wnd := FMain.Handle;

uID := 1;

uFlags := NIF\_ICON or NIF\_MESSAGE or NIF\_TIP;

hicon := Icon.Handle;

uCallbackMessage := WM\_USER + 1;

szTip := 'Программа отправки данных в головное отделение';

End;

Case n OF

1:

Shell\_NotifyIcon(Nim\_Add, @Nim);

2:

Shell\_NotifyIcon(Nim\_Delete, @Nim);

3:

Shell\_NotifyIcon(Nim\_Modify, @Nim);

End;

end;

end.

Текст программы LinkUp Настройки

unit Uftpsett;

interface

uses

Winapi.Windows, Winapi.Messages, System.SysUtils, System.Variants, System.Classes, Vcl.Graphics,

Vcl.Controls, Vcl.Forms, Vcl.Dialogs, Vcl.StdCtrls, inifiles;

type

TFFtpsett = class(TForm)

GroupBox1: TGroupBox;

Button1: TButton;

Label1: TLabel;

Label2: TLabel;

Label3: TLabel;

Edit1: TEdit;

Edit2: TEdit;

Edit3: TEdit;

procedure Button1Click(Sender: TObject);

procedure FormCreate(Sender: TObject);

private

{ Private declarations }

public

{ Public declarations }

end;

var

FFtpsett: TFFtpsett;

implementation

{$R \*.dfm}

procedure TFFtpsett.Button1Click(Sender: TObject);

var

ini: tinifile;

begin

ini := tinifile.Create(Extractfilepath(Application.ExeName) + '\' +

'link.ini');

ini.WriteString('REMOTE', 'FTPuser', Edit2.Text);

ini.WriteString('REMOTE', 'FTPpass', Edit3.Text);

ini.WriteString('REMOTE', 'FTPserv', Edit1.Text);

ini.free;

end;

procedure TFFtpsett.FormCreate(Sender: TObject);

var

ini: tinifile;

begin

ini := tinifile.Create(Extractfilepath(Application.ExeName) + '\' +

'link.ini');

Edit2.Text := ini.ReadString('REMOTE', 'FTPuser', '');

Edit3.Text := ini.ReadString('REMOTE', 'FTPpass', '');

Edit1.Text := ini.ReadString('REMOTE', 'FTPserv', '');

ini.free;

end;

end.

Текст программы LinkDown

unit UMainLink;

interface

uses

Winapi.Windows, Winapi.Messages, System.SysUtils, System.Variants,

System.Classes, Vcl.Graphics,

Vcl.Controls, Vcl.Forms, Vcl.Dialogs, IdBaseComponent, IdComponent,

IdTCPConnection, IdTCPClient, IdExplicitTLSClientServerBase, IdFTP,

Vcl.ExtCtrls, Vcl.Menus, Vcl.StdCtrls, inifiles, ustr, ShellAPI, IOUtils,

FileCtrl,

FireDAC.Stan.Intf, FireDAC.Stan.Option, FireDAC.Stan.Error, FireDAC.UI.Intf,

FireDAC.Phys.Intf, FireDAC.Stan.Def, FireDAC.Stan.Pool, FireDAC.Stan.Async,

FireDAC.Phys, FireDAC.Phys.Oracle, FireDAC.Phys.OracleDef,

FireDAC.VCLUI.Wait,

FireDAC.Stan.Param, FireDAC.DatS, FireDAC.DApt.Intf, FireDAC.DApt, Data.DB,

FireDAC.Comp.DataSet, DateUtils, FireDAC.Comp.Client;

type

TFMainLink = class(TForm)

Label3: TLabel;

Label4: TLabel;

GroupBox1: TGroupBox;

EditInFolder: TEdit;

BSelectLocalFolder: TButton;

GroupBox3: TGroupBox;

Label1: TLabel;

Label2: TLabel;

Edit4: TEdit;

CheckBox1: TCheckBox;

Edit5: TEdit;

BSelectArcFolder: TButton;

BSaveSettings: TButton;

PopupMenu1: TPopupMenu;

N1: TMenuItem;

N2: TMenuItem;

Timer1: TTimer;

FDC: TFDConnection;

Query1: TFDQuery;

procedure BSaveSettingsClick(Sender: TObject);

procedure FormClose(Sender: TObject; var Action: TCloseAction);

procedure FormCreate(Sender: TObject);

Procedure LoadIni;

Procedure Ic(n: Integer; Icon: TIcon);

Procedure Log(f: string);

procedure N1Click(Sender: TObject);

procedure N2Click(Sender: TObject);

procedure Timer1Timer(Sender: TObject);

procedure Button1Click(Sender: TObject);

procedure FormCloseQuery(Sender: TObject; var CanClose: Boolean);

private

Procedure OnMinimizeProc(Sender: TObject);

public

{ Public declarations }

protected

Procedure ControlWindow(Var Msg: TMessage); message WM\_SYSCOMMAND;

Procedure IconMouse(var Msg: TMessage); message WM\_USER + 1;

end;

var

FMainLink: TFMainLink;

implementation

{$R \*.dfm}

procedure TFMainLink.Button1Click(Sender: TObject);

begin

Timer1.Enabled := Not Timer1.Enabled;

end;

Procedure TFMainLink.ControlWindow(Var Msg: TMessage);

Begin

IF Msg.WParam = SC\_MINIMIZE then

Begin

Ic(1, Application.Icon); // Добавляем значок в трей

ShowWindow(Handle, SW\_HIDE); // Скрываем программу

ShowWindow(Application.Handle, SW\_HIDE); // Скрываем кнопку с TaskBar'а

End

else

inherited;

End;

procedure TFMainLink.IconMouse(var Msg: TMessage);

Var

p: tpoint;

begin

GetCursorPos(p); // Запоминаем координаты курсора мыши

Case Msg.LParam OF // Проверяем какая кнопка была нажата

WM\_LBUTTONUP, WM\_LBUTTONDBLCLK

: { Действия, выполняемый по одинарному или двойному щелчку левой кнопки мыши на значке. В нашем случае это просто активация приложения }

Begin

Ic(2, Application.Icon); // Удаляем значок из трея

ShowWindow(Application.Handle, SW\_SHOW);

// Восстанавливаем кнопку программы

ShowWindow(Handle, SW\_SHOW); // Восстанавливаем окно программы

End;

WM\_RBUTTONUP

: { Действия, выполняемый по одинарному щелчку правой кнопки мыши }

Begin

SetForegroundWindow(Handle);

// Восстанавливаем программу в качестве переднего окна

PopupMenu1.Popup(p.X, p.Y);

// Заставляем всплыть тот самый TPopUp о котором я говорил чуть раньше

PostMessage(Handle, WM\_NULL, 0, 0);

end;

End;

end;

Procedure TFMainLink.Log(f: string);

begin

// логируем

Label4.Caption := f;

end;

Procedure TFMainLink.LoadIni;

var

ini: tinifile;

begin

ini := tinifile.Create(Extractfilepath(Application.ExeName) + '\' +

'linkdown.ini');

try

Timer1.Interval := ini.ReadInteger('INTERVAL', 'mSec', 20000);

except

on E: exception do

begin

Timer1.Interval := 20000;

ShowMessage('В настройках нужно указать время таймера.');

E := Nil;

end;

end;

EditInFolder.Text := ini.ReadString('LOCAL', 'Path', '');

Edit5.Text := ini.ReadString('LOCAL', 'Archive', '');

CheckBox1.checked := ini.ReadBool('SYNC', 'Enabled', True);

Timer1.Enabled := CheckBox1.checked;

Edit4.Text := Inttostr(Round(Timer1.Interval / 1000));

ini.free;

end;

procedure TFMainLink.N1Click(Sender: TObject);

begin

Begin

Ic(2, Application.Icon); // Удаляем значок из трея

ShowWindow(Application.Handle, SW\_SHOW);

// Восстанавливаем кнопку программы

ShowWindow(Handle, SW\_SHOW); // Восстанавливаем окно программы

End;

end;

procedure TFMainLink.N2Click(Sender: TObject);

begin

Application.Terminate;

end;

procedure TFMainLink.FormClose(Sender: TObject; var Action: TCloseAction);

begin

// а то задолбает вусмерть

if MessageDlg('Вы действительно ходите закрыть?', mtInformation, [mbYes, mbNo] ,0) = mrYes

then Application.Terminate;

Action := caNone;

//canClose:=false;

Application.Minimize;

end;

procedure TFMainLink.FormCloseQuery(Sender: TObject; var CanClose: Boolean);

begin

// if MessageDlg('Вы действительно ходите закрыть?', mtInformation, [mbYes, mbNo] ,0) = mrYes

// then Application.Terminate;

end;

procedure TFMainLink.FormCreate(Sender: TObject);

begin

Application.onMinimize := OnMinimizeProc;

LoadIni;

// errorcount := 0;

// arcpath := Edit5.Text + '\' + Inttostr(YearOf(date)) + '\' +

// Inttostr(MonthOfTheYear(date)) + '\' + Inttostr(DayOfTheMonth(date)) + '\';

// CreateDirEx(arcPath);

end;

Procedure TFMainLink.OnMinimizeProc(Sender: TObject);

Begin

PostMessage(Handle, WM\_SYSCOMMAND, SC\_MINIMIZE, 0);

End;

// -------------------------------------------------------------------------------------------------------

procedure TFMainLink.Timer1Timer(Sender: TObject);

var

filename, fn, pth, nfn,ts: string;

sl: tstringlist;

i, j: Integer;

org, ndoc, fdoc, nfiles, dt, tm, fdt, ftm, mon, year, dop1, dop2: string;

begin

if not CheckBox1.checked then

exit;

Log('Начата обработка');

// тут и будет основная работа, а именно - подгрузить обработчики dscr

sl := tstringlist.Create;

for filename in TDirectory.GetFiles(EditInFolder.Text, '\*.dscr') do

begin

sl.LoadFromFile(filename);

fdoc := sl.Strings[0];

org := sl.Strings[1];

ndoc := sl.Strings[2];

ts:= sl.Strings[3];

delete(ts,1,2);

delete(ts,3,2);

mon:=ts;

ts:= sl.Strings[3];

delete(ts,1,4);

year:='20'+ts;

fdt := sl.Strings[4];

ftm := sl.Strings[5];

pth := Inttostr(YearOf(Date)) + '\' +

Inttostr(MonthOfTheYear(Date)) + '\' + Inttostr(DayOfTheMonth(Date));

fn := Edit5.Text + '\' + pth;

Createdirex(fn);

tm := Timetostr(time);

j := 0;

for i := 6 to sl.Count - 1 do

begin

j := j + 1;

nfn := fn + '\' + sl.Strings[i];

DeleteFile(nfn);

Movefile(PChar(EditInFolder.Text + '\' + sl.Strings[i]),

PChar(nfn));

Query1.sql.Clear;

Query1.sql.add('INSERT INTO VCM.FILES (FNAME, FDOC) VALUES ( ' +

Quotedstr(nfn) + ' , ' + Quotedstr(fdoc) + ' )');

Query1.ExecSQL;

end;

nfiles := Inttostr(j);

deletefile(fn + '\' + extractfilename(filename));

Movefile(PChar(filename), PChar(fn + '\' + extractfilename(filename)));

// сделать запись в документах

if j<>strtoint(nfiles) then dop1:='Обнаружено несовпадение количества файлов';

Query1.sql.Clear;

Query1.sql.add('insert into doc\_del select \* from doc where fdoc='+Quotedstr(fdoc));

Query1.ExecSQL;

Query1.sql.Clear;

Query1.sql.add('DELETE FROM DOC WHERE fdoc='+Quotedstr(fdoc));

Query1.ExecSQL;

Query1.sql.Clear;

Query1.sql.add

('INSERT INTO DOC (ORG, NDOC,FDOC, NFILES, TM, MON, YEAR,');

Query1.sql.add('DOP1, DOP2, FDT, FTM, RFILES)VALUES (' + org + ', ' +ndoc + ', ');

Query1.sql.add(Quotedstr(fdoc) + ', ' + nfiles + ', ');

Query1.sql.add(Quotedstr(tm) + ', ' + mon + ', ');

Query1.sql.add(year + ', ' + Quotedstr(dop1) + ', ');

Query1.sql.add(Quotedstr(dop2) + ', ' + Quotedstr(fdt) + ', ' +

Quotedstr(ftm) + ','+inttostr(j)+')');

//Query1.sql.SaveToFile('s:\2\qq');

Query1.ExecSQL;

Query1.sql.Clear;

end;

sl.free;

Log('Ожидаю файлов...');

end;

// -------------------------------------------------------------------------------------------------------

procedure TFMainLink.BSaveSettingsClick(Sender: TObject);

var

ini: tinifile;

begin

ini := tinifile.Create(Extractfilepath(Application.ExeName) + '\' +

'linkdown.ini');

try

Timer1.Interval := Strtoint(Edit4.Text) \* 1000;

Except

on E: exception do

begin

ini.free;

ShowMessage

('В настройках таймера допускается только целое число секунд.');

E := Nil;

exit;

end;

end;

ini.WriteInteger('INTERVAL', 'mSec', Timer1.Interval);

ini.WriteString('LOCAL', 'Path', EditInFolder.Text);

ini.WriteString('LOCAL', 'Archive', Edit5.Text);

ini.WriteBool('SYNC', 'Enabled', CheckBox1.checked);

ini.free;

end;

Procedure TFMainLink.Ic(n: Integer; Icon: TIcon);

Var

Nim: TNotifyIconData;

begin

With Nim do

Begin

cbSize := Nim.SizeOf();

Wnd := FMainLink.Handle;

uID := 1;

uFlags := NIF\_ICON or NIF\_MESSAGE or NIF\_TIP;

hicon := Icon.Handle;

uCallbackMessage := WM\_USER + 1;

szTip := 'Программа загрузки данных из отделений';

End;

Case n OF

1:

Shell\_NotifyIcon(Nim\_Add, @Nim);

2:

Shell\_NotifyIcon(Nim\_Delete, @Nim);

3:

Shell\_NotifyIcon(Nim\_Modify, @Nim);

End;

end;

end.

Текст программы бухгалтерской части

unit UMainMain;

interface

uses

Winapi.Windows, Winapi.Messages, System.SysUtils, System.Variants, System.Classes, Vcl.Graphics,

Vcl.Controls, Vcl.Forms, Vcl.Dialogs, Vcl.ExtCtrls, Vcl.StdCtrls,

Vcl.Samples.Spin,PdfiumCore,PdfiumCtrl, Vcl.Buttons, FireDAC.Stan.Intf,

FireDAC.Stan.Option, FireDAC.Stan.Error, FireDAC.UI.Intf, FireDAC.Phys.Intf,

FireDAC.Stan.Def, FireDAC.Stan.Pool, FireDAC.Stan.Async, FireDAC.Phys,

FireDAC.Phys.Oracle, FireDAC.Phys.OracleDef, FireDAC.VCLUI.Wait,

FireDAC.Stan.Param, FireDAC.DatS, FireDAC.DApt.Intf, FireDAC.DApt, Data.DB,

FireDAC.Comp.DataSet, FireDAC.Comp.Client,ShellAPI,IOUtils, Vcl.Grids, Vcl.ComCtrls, ustr;

type

TForm3 = class(TForm)

Panel1: TPanel;

Panel2: TPanel;

PrintDialog1: TPrintDialog;

Panel3: TPanel;

Panel4: TPanel;

btnScale: TButton;

chkLCDOptimize: TCheckBox;

chkSmoothScroll: TCheckBox;

edtZoom: TSpinEdit;

chkChangePageOnMouseScrolling: TCheckBox;

FDC: TFDConnection;

QFind: TFDQuery;

QFindf: TFDQuery;

BitBtn2: TBitBtn;

Splitter1: TSplitter;

Panel5: TPanel;

StringGrid2: TStringGrid;

Label2: TLabel;

Splitter2: TSplitter;

Panel6: TPanel;

StringGrid1: TStringGrid;

Edit1: TEdit;

Label1: TLabel;

GroupBox1: TGroupBox;

Label3: TLabel;

Label4: TLabel;

Label5: TLabel;

ComboBox1: TComboBox;

Edit2: TEdit;

Edit3: TEdit;

Qtemp: TFDQuery;

Button1: TButton;

Label6: TLabel;

Label7: TLabel;

Label8: TLabel;

Label9: TLabel;

Label10: TLabel;

Label11: TLabel;

Label12: TLabel;

Label13: TLabel;

Label14: TLabel;

Label15: TLabel;

Label16: TLabel;

Label17: TLabel;

Label18: TLabel;

Label19: TLabel;

Button2: TButton;

BitBtn1: TBitBtn;

BitBtn3: TBitBtn;

Qcorr: TFDQuery;

qcorrupd: TFDQuery;

Label20: TLabel;

Label21: TLabel;

DateTimePicker1: TDateTimePicker;

DateTimePicker2: TDateTimePicker;

Button3: TButton;

GroupBox2: TGroupBox;

qfinddat: TFDQuery;

procedure btnScaleClick(Sender: TObject);

procedure FormCreate(Sender: TObject);

procedure BitBtn1Click(Sender: TObject);

procedure edtZoomChange(Sender: TObject);

procedure chkSmoothScrollClick(Sender: TObject);

procedure chkLCDOptimizeClick(Sender: TObject);

procedure chkChangePageOnMouseScrollingClick(Sender: TObject);

procedure Edit1Change(Sender: TObject);

procedure StringGrid1Click(Sender: TObject);

procedure StringGrid2Click(Sender: TObject);

procedure BitBtn2Click(Sender: TObject);

procedure Button1Click(Sender: TObject);

procedure Button2Click(Sender: TObject);

procedure BitBtn3Click(Sender: TObject);

procedure Button3Click(Sender: TObject);

private

FCtrl: TPdfControl;

procedure PrintDocument(Sender: TObject);

public

procedure fillfind(Stringgrid1:TStringGrid;QFind:TFDQuery;s:string);

procedure clearstring(Stringgrid1:TStringgrid);

end;

var

Form3: TForm3;

implementation

uses

System.TypInfo, Vcl.Printers, USettmain;

{$R \*.dfm}

procedure TForm3.PrintDocument(Sender: TObject);

begin

TPdfDocumentVclPrinter.PrintDocument(FCtrl.Document, ExtractFileName(FCtrl.Document.FileName));

end;

procedure TForm3.StringGrid1Click(Sender: TObject);

var

s:string;

begin

QTemp.Close;

QTemp.SQL.Clear;

QTemp.SQL.Add('select \* from doc where fdoc='+QuotedStr(stringgrid1.cells[0,stringgrid1.row]));

QTemp.Open;

Label19.caption:=stringgrid1.cells[0,stringgrid1.row];

Label13.caption:=QTemp.FieldByName('nfiles').Asstring;

Label14.caption:=QTemp.FieldByName('rfiles').Asstring;

Label15.caption:=QTemp.FieldByName('fdt').Asstring;

Label16.caption:=QTemp.FieldByName('dt').Asstring;

Label17.caption:=QTemp.FieldByName('ftm').Asstring;

Label18.caption:=QTemp.FieldByName('tm').Asstring;

s:=stringgrid1.cells[0,stringgrid1.row];

QFindf.close;

QFindf.ParamByName('f').asstring:=s;

Qfindf.Open;

fillfind(stringgrid2, qfindf, 'fname');

end;

procedure TForm3.StringGrid2Click(Sender: TObject);

var

s:string;

begin

try

s:=stringgrid2.cells[0,stringgrid2.row];

//Panel1.Caption:=s;

FCtrl.LoadFromFile(s);

FCtrl.ScaleMode := smZoom;

except

On E:exception do

begin

ShowMessage('Файл не найден в архиве и не может быть отображен!');

E:=Nil;

end;

end;

end;

procedure TForm3.BitBtn1Click(Sender: TObject);

begin

// FCtrl.LoadFromFile('s:\3\2023\12\31\5500007-311223-14.pdf');

Form4.ShowModal;

end;

procedure TForm3.BitBtn2Click(Sender: TObject);

begin

FCtrl.PrintDocument;

end;

procedure TForm3.BitBtn3Click(Sender: TObject);

var

fld:string;

doc,filename:string;

s,p,fname:string;

begin

//тут мы проводим корректировку из папки

//старые файлы текущего документа переименовать в папке и

//в базе, новые внести в базу и скопировать 5500003-030124

if MessageDlg('Вы действительно ходите провести корректировку текущего документа?',

mtInformation, [mbYes, mbNo] ,0) = mrYes

then

begin

if label19.caption='0000000-000000' then

begin

ShowMessage('Не выбран ни один документ');

Exit;

end;

fld:=form4.edit1.text;

doc:=label19.caption;

//delete(doc,3,1);

//showmessage(doc);

qcorr.close;

qcorr.ParamByName('fdoc').asstring:=doc;

QCorr.Open;

QCorr.first;

while not qcorr.eof do

begin

//update files set fname=:fname where fdoc=:fdoc

s:=qcorr.fieldbyname('fname').Asstring;

p:=extractfilepath(s);

fname:=p+'corr\_'+extractfilename(s);

//showmessage(fname);

qcorrupd.ParamByName('fdoc').asstring:=doc;

qcorrupd.ParamByName('fname').asstring:=fname;

qcorrupd.ExecSQL;

renamefile(qcorr.fieldbyname('fname').Asstring,fname);

qcorr.next;

end;

CREATEDIREX(P);

for filename in TDirectory.GetFiles(fld, '\*.pdf') do

begin

DeleteFile(fname);

Movefile(PChar(filename),PChar(p+doc+ '\_'+extractfilename(filename)));

QTemp.Close;

QTemp.sql.Clear;

QTemp.sql.add('INSERT INTO VCM.FILES (FNAME, FDOC) VALUES ( ' +

Quotedstr(p+doc+'\_'+ extractfilename(filename)) + ' , ' + Quotedstr(doc) + ' )');

QTemp.ExecSQL;

end;

ShowMessage('Корректировка выполнена');

end;

end;

procedure TForm3.btnScaleClick(Sender: TObject);

begin

if FCtrl.ScaleMode = High(FCtrl.ScaleMode) then

FCtrl.ScaleMode := Low(FCtrl.ScaleMode)

else

FCtrl.ScaleMode := Succ(FCtrl.ScaleMode);

Caption := GetEnumName(TypeInfo(TPdfControlScaleMode), Ord(FCtrl.ScaleMode));

end;

procedure TForm3.Button1Click(Sender: TObject);

var

num:integer;

s:string;

begin

if ComboBox1.Text<>'' then

begin

QTemp.Close;

QTemp.SQL.Clear;

QTemp.SQL.Add('select num from org where name='+Quotedstr(ComboBox1.Text));

QTemp.Open;

num:=QTemp.FieldByName('num').AsInteger;

QTemp.Close;

end;

QTemp.Close;

QTemp.SQL.Clear;

QTemp.SQL.Add('select \* from doc where 1=1 ');

if ComboBox1.Text<>'' then

begin

QTemp.SQL.Add(' and org='+Inttostr(num)+' ');

end;

if Edit2.Text<>'' then

begin

QTemp.SQL.Add(' and ndoc='+Edit2.text+' ');

end;

if Edit3.Text<>'' then

begin

//QTemp.SQL.Add(' and fdoc like '+Quotedstr('%'+Edit3.text));

s:= Edit3.text;

QTemp.SQL.Add(' and dt = '+Quotedstr(s));

end;

QTemp.Open;

fillfind(Stringgrid1,Qtemp, 'fdoc');

end;

procedure TForm3.Button2Click(Sender: TObject);

begin

ComboBox1.Text:='';

Edit2.Text:='';

Edit3.Text:='';

end;

procedure TForm3.Button3Click(Sender: TObject);

begin

qfinddat.close;

qfinddat.parambyname('d1').asdate:=datetimepicker1.date;

qfinddat.parambyname('d2').asdate:=datetimepicker2.date;

Qfinddat.open;

fillfind(Stringgrid1,Qfinddat, 'fdoc');

end;

procedure TForm3.chkChangePageOnMouseScrollingClick(Sender: TObject);

begin

FCtrl.ChangePageOnMouseScrolling := chkChangePageOnMouseScrolling.Checked;

end;

procedure TForm3.chkLCDOptimizeClick(Sender: TObject);

begin

if chkLCDOptimize.Checked then

FCtrl.DrawOptions := FCtrl.DrawOptions + [proLCDOptimized]

else

FCtrl.DrawOptions := FCtrl.DrawOptions - [proLCDOptimized];

end;

procedure TForm3.chkSmoothScrollClick(Sender: TObject);

begin

FCtrl.SmoothScroll := chkSmoothScroll.Checked;

end;

procedure TForm3.clearstring(Stringgrid1:TStringGrid);

var

i,j:integer;

begin

for i:=1 to stringgrid1.rowcount-1 do

for j:=1 to stringgrid1.colcount-1 do

Stringgrid1.cells[j,i]:='';

Stringgrid1.rowcount:=2;

end;

procedure TForm3.fillfind(Stringgrid1:TStringGrid;QFind:TFDQuery;s:string);

var

i:integer;

begin

ClearString(Stringgrid1);

i:=1;

while not QFind.eof do

begin

Stringgrid1.cells[0,i]:=QFind.FieldByName(s).Asstring;

QFind.Next;

i:=i+1;

Stringgrid1.rowcount:=Stringgrid1.rowcount+1;

end;

Stringgrid1.rowcount:=Stringgrid1.rowcount-1;

end;

procedure TForm3.Edit1Change(Sender: TObject);

begin

qfind.close;

qfind.ParamByName('f').AsString:='%'+edit1.text+'%';

qfind.open;

fillfind(Stringgrid1,Qfind, 'fdoc');

end;

procedure TForm3.edtZoomChange(Sender: TObject);

begin

FCtrl.ZoomPercentage := edtZoom.Value;

end;

procedure TForm3.FormCreate(Sender: TObject);

begin

PDFiumDllDir := ExtractFilePath(ParamStr(0));

datetimepicker1.date:=date;

datetimepicker2.date:=date;

FCtrl := TPdfControl.Create(Self);

FCtrl.Align := alClient;

FCtrl.Parent := Self;

//FCtrl.SendToBack;

FCtrl.Color := clGray;

FCtrl.ScaleMode := smFitWidth;

FCtrl.OnPrintDocument := PrintDocument;

edtZoom.Value := FCtrl.ZoomPercentage;

QTemp.SQL.add('select distinct(name) n from org');

QTemp.Open;

ComboBox1.Text:='';

while not QTemp.eof do

begin

ComboBox1.Items.Add(QTemp.FieldByNAme('n').Asstring);

QTemp.Next;

end;

QTemp.Close;

end;

end.

Текст программы бухгалтерской части – настройки.

unit USettmain;

interface

uses

Winapi.Windows, Winapi.Messages, System.SysUtils, System.Variants, System.Classes, Vcl.Graphics,

Vcl.Controls, Vcl.Forms, Vcl.Dialogs, Vcl.StdCtrls,FileCtrl, inifiles,

Data.DB, FireDAC.Stan.Intf, FireDAC.Stan.Option, FireDAC.Stan.Param,

FireDAC.Stan.Error, FireDAC.DatS, FireDAC.Phys.Intf, FireDAC.DApt.Intf,

FireDAC.Stan.Async, FireDAC.DApt, FireDAC.Comp.DataSet, FireDAC.Comp.Client,

Vcl.Grids, Vcl.DBGrids;

type

TForm4 = class(TForm)

GroupBox1: TGroupBox;

Edit1: TEdit;

Button1: TButton;

Button2: TButton;

GroupBox2: TGroupBox;

DBGrid1: TDBGrid;

DataSource1: TDataSource;

FDTable1: TFDTable;

procedure Button2Click(Sender: TObject);

procedure FormCreate(Sender: TObject);

procedure Button1Click(Sender: TObject);

private

{ Private declarations }

public

{ Public declarations }

end;

var

Form4: TForm4;

implementation

uses umainmain;

{$R \*.dfm}

procedure TForm4.Button1Click(Sender: TObject);

var

SelectedFolder: string;

begin

SelectDirectory('Выберите директорию', '', SelectedFolder);

Edit1.Text := SelectedFolder;

end;

procedure TForm4.Button2Click(Sender: TObject);

var

ini:tinifile;

s:string;

begin

s:= extractfilepath(application.exename)+'Main.ini';

ini:=tinifile.create(s);

ini.writestring('FOLDERS','correctionfolder',edit1.text);

ini.free;

with form3 do

begin

QTemp.close;

QTemp.SQL.Clear;

QTemp.SQL.add('select distinct(name) n from org');

QTemp.Open;

ComboBox1.Text:='';

ComboBox1.items.Clear;

while not QTemp.eof do

begin

ComboBox1.Items.Add(QTemp.FieldByNAme('n').Asstring);

QTemp.Next;

end;

QTemp.Close;

end;

Form4.Close;

end;

procedure TForm4.FormCreate(Sender: TObject);

var

s:string;

ini:tinifile;

begin

s:= extractfilepath(application.exename)+'Main.ini';

ini:=tinifile.create(s);

edit1.text:=ini.Readstring('FOLDERS','correctionfolder','');

ini.free;

Form4.Close;

end;

end.

Текст программы операторской части

unit UMain;

interface

uses

Winapi.Windows, Winapi.Messages, System.SysUtils, System.Variants,

System.Classes, Vcl.Graphics,

Vcl.Controls, Vcl.Forms, Vcl.Dialogs, Vcl.ExtCtrls, Vcl.StdCtrls,

Vcl.ComCtrls, inifiles,

Vcl.Buttons, Vcl.Imaging.jpeg, IOUtils, FileCtrl,

DateUtils, ustr, FireDAC.Stan.Intf, FireDAC.Stan.Option, FireDAC.Stan.Error,

FireDAC.UI.Intf, FireDAC.Phys.Intf, FireDAC.Stan.Def, FireDAC.Stan.Pool,

FireDAC.Stan.Async, FireDAC.Phys, FireDAC.VCLUI.Wait, FireDAC.Stan.Param,

FireDAC.DatS, FireDAC.DApt.Intf, FireDAC.DApt, Data.DB,

FireDAC.Comp.DataSet,

FireDAC.Comp.Client, FireDAC.Phys.SQLite, FireDAC.Phys.SQLiteDef,

FireDAC.Stan.ExprFuncs, TlHelp32,FireDAC.Phys.SQLiteWrapper.Stat;

type

TForm1 = class(TForm)

DateTimePicker1: TDateTimePicker;

Label1: TLabel;

Edit1: TEdit;

Label2: TLabel;

Edit2: TEdit;

BtnScanClick: TBitBtn;

Label3: TLabel;

Label4: TLabel;

BtnRecord: TBitBtn;

BitBtn3: TBitBtn;

BitBtn4: TBitBtn;

Timer1: TTimer;

GroupBox1: TGroupBox;

Label5: TLabel;

Label6: TLabel;

EditInFolder: TEdit;

BSelectInFolder: TButton;

EditProcess: TEdit;

CheckBoxForRun: TCheckBox;

GroupBox2: TGroupBox;

Label7: TLabel;

EditOutFolder: TEdit;

BSelectOutFolder: TButton;

ButtonSaveSettings: TBitBtn;

FDC: TFDConnection;

Query1: TFDQuery;

Edit3: TEdit;

Button1: TButton;

OpenDialog1: TOpenDialog;

Label8: TLabel;

procedure BitBtn3Click(Sender: TObject);

procedure BitBtn4Click(Sender: TObject);

procedure FormCreate(Sender: TObject);

procedure BtnRecordClick(Sender: TObject);

procedure Timer1Timer(Sender: TObject);

procedure ButtonSaveSettingsClick(Sender: TObject);

procedure Button1Click(Sender: TObject);

procedure FormCloseQuery(Sender: TObject; var CanClose: Boolean);

private

{ Private declarations }

public

procedure MakeFolders;

procedure SaveSettings;

function runned(S:string):boolean;

procedure runprocess(S:string);

end;

var

Form1: TForm1;

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

basedir: string;

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

implementation

{$R \*.dfm}

uses Unit2, Usett;

procedure TForm1.MakeFolders;

begin

basedir := ExtractFilePath(ParamStr(0));

try

CreateDirEx(EditInFolder.Text);

CreateDirEx(EditOutFolder.Text);

Except

on E:exception do

begin

ShowMessage('Нужно проверить настройки папок');

Timer1.Enabled:=false;

E:=Nil;

end;

end;

// CreateDirEx(Edit4.text);

end;

function TForm1.runned(S:string):boolean;

var SnapProc:THandle;

ProcEntry:TProcessEntry32;

Item:TListItem;

begin

//CLearAll;

// Создаём снимок, в котором сохраняем все процессы, а

// затем в цикле получаем информацию о кажом из этих

// процессов, перенося её в ListProcesses

SnapProc:=CreateToolhelp32Snapshot(TH32CS\_SnapProcess,0);

runned:=false;

if SnapProc<>Invalid\_Handle\_Value then

try

ProcEntry.dwSize:=SizeOf(TProcessEntry32);

if Process32First(SnapProc,ProcEntry) then

repeat

//Item:=ListProcesses.Items.Add;

if editprocess.text=ProcEntry.szExeFile then

begin

runned:=true;

label8.caption:= editprocess.text;

end;

//Item.SubItems.Add(IntToStr(ProcEntry.th32ProcessID));

//Item.SubItems.Add(IntToStr(ProcEntry.th32ParentProcessID));

//Item.SubItems.Add(IntToStr(ProcEntry.cntThreads));

// Сохраняем PID в поле Data соответствующего

// элемента списка. Вообще, поле Data имеет тип

// Pointer, а PID - это целое число, но т.к. оба этих

// типа 32-битные, их можно приводить друг к другу

//Item.Data:=Pointer(ProcEntry.th32ProcessID)

until not Process32Next(SnapProc,ProcEntry)

finally

CloseHandle(SnapProc)

end

else

begin

//ListProcesses.Visible:=False;

ShowMessage('Невозможно получить список процессов:'#13#10'Ошибка №'+IntToStr(GetLastError))

end;

end;

procedure TForm1.runprocess(S:string);

var

ts:string;

begin

startprog(Handle,edit3.text,'');

end;

procedure TForm1.Timer1Timer(Sender: TObject);

var

FileName: string;

i: integer;

begin

DateTimePicker1.Date := Date;

i := 0;

for FileName in TDirectory.GetFiles(EditInFolder.Text, '\*.\*') do

begin

i := i + 1;

end;

Label4.caption := InttoStr(i);

if not runned(EDitprocess.text) then

if checkboxforrun.checked then runprocess(EDit3.text);

end;

procedure TForm1.BtnRecordClick(Sender: TObject);

var

FileName: string;

ndoc, nfiles: integer;

fdoc, nname, t: string;

s, mon, year, tm, dt, org, nndoc: string;

buttonSelected, i: integer;

sl:tstringlist;

begin

// в папке камеры лежат файлы, необходимо создать документ

// в нем номер, имена файлов, имя файла документа это его номер

// файлы и документ сложить в папку исходящих откуда их заберет

// программа синхронизации, она закачает их и перенесет в архив

// исходные файлы нужно удалить

// по желанию сделать запись в локальной базе данных

// файлы камеры переименовать так же по номеру документа и порядку

// не забываем проверить на существование и если надо - перезаписать

if Label4.Caption = '0' then

begin

ShowMessage('В папке входящих не обнаружено файлов для добавления');

exit;

end;

FDC.params.Database := ExtractFilePath(Application.ExeName) + 'VCM.SQLITE';

FDC.Open();

Query1.SQL.Clear;

while length(Edit1.Text)<5 do Edit1.Text:= '0' +Edit1.Text;

fdoc := Label1.caption + Edit1.Text + '-' + Edit2.Text;

ndoc := Strtoint(Edit1.Text);

s := Edit2.Text;

delete(s, 1, 2);

delete(s, 3, 2);

mon := s;

s := Edit2.Text;

delete(s, 1, 4);

year := s;

nfiles := Strtoint(Label4.caption);

tm := TimeToStr(time);

dt := Datetostr(Date);

org := Label1.caption;

Query1.Open('select fdoc from doc where fdoc=' + Quotedstr(fdoc));

if Query1.FieldByName('fdoc').Asstring = fdoc then

begin

buttonSelected :=

MessageDlg('Такой документ уже существует, перезаписать?', mtError,

mbYesNo, 0);

if buttonSelected = mrYes then

begin

Query1.SQL.Clear;

Query1.SQL.Add('delete from doc where fdoc=' + Quotedstr(fdoc));

Query1.ExecSQL;

end

else

begin

// формируем новый номер документа для следующего

Query1.Open('select max(ndoc)+1 n from doc');

nndoc := Query1.FieldByName('n').Asstring;

while length(nndoc) < 5 do

nndoc := '0' + nndoc;

Edit1.Text := nndoc;

Exit;

end;

end;

i:=0;

sl:=Tstringlist.Create;

sl.add(fdoc);

sl.add(Label1.Caption);

sl.add(Edit1.Text);

sl.add(Edit2.Text);

sl.add(Datetostr(Date));

sl.add(Timetostr(Time));

for FileName in TDirectory.GetFiles(EditInFolder.Text, '\*.\*') do

begin

// положить в Fsett.Edit1.text файлы и описатель

// и удалить

i:=i+1;

t:= extractfilename(filename)+extractfileext(filename);

nname:=EditOutFolder.Text+'\'+fdoc+'-'+inttostr(i)+extractfileext(filename);

if Not MoveFile(PChar(filename), PChar(nname)) then

begin

i:=i-1;

end

else sl.add(extractfilename(nname));

end;

nfiles:=i;

// описатель

sl.savetofile(EditOutFolder.Text+'\'+fdoc+'.dscr');

sl.free;

Query1.SQL.Clear;

Query1.SQL.Add

('INSERT INTO doc (org,ndoc,fdoc,nfiles,dt,tm,mon, year) VALUES (');

Query1.SQL.Add(org + ',' + InttoStr(ndoc) + ',' + Quotedstr(fdoc) + ',' +

InttoStr(nfiles) + ',' + Quotedstr(dt));

Query1.SQL.Add(',' + Quotedstr(tm) + ',' + Quotedstr(mon) + ',' +

Quotedstr(year) + ')');

Query1.ExecSQL;

// формируем новый номер документа для следующего

Query1.Open('select max(ndoc)+1 n from doc');

nndoc := Query1.FieldByName('n').Asstring;

while length(nndoc) < 5 do

nndoc := '0' + nndoc;

Edit1.Text := nndoc;

FDC.Close;

end;

procedure TForm1.Button1Click(Sender: TObject);

begin

if OpeNDialog1.Execute() then

EDit3.text:=OpenDialog1.Filename;

end;

procedure TForm1.ButtonSaveSettingsClick(Sender: TObject);

begin

SaveSettings;

end;

procedure TForm1.SaveSettings;

begin

ini := Tinifile.Create(ExtractFilePath(ParamStr(0)) + 'VCM.ini');

ini.WriteString('FOLDERS', 'Input', EditInFolder.Text);

ini.WriteString('FOLDERS', 'Output', EditOutFolder.Text);

ini.WriteString('CAMERA', 'Process', EditProcess.Text);

ini.WriteString('ORG', 'Num', Label1.caption);

ini.WriteBool('CAMERA', 'RunProcess', CheckBoxForRun.Checked);

ini.WriteString('CAMERA', 'RunApp', Edit3.Text);

ini.free;

end;

procedure TForm1.BitBtn3Click(Sender: TObject);

begin

if MessageDlg('Вы действительно ходите закрыть?', mtInformation, [mbYes, mbNo] ,0) = mrYes

then Application.Terminate;

end;

procedure TForm1.BitBtn4Click(Sender: TObject);

begin

if Form1.Height = 41 then

Form1.Height := 138

else

Form1.Height := 41;

end;

procedure TForm1.FormCloseQuery(Sender: TObject; var CanClose: Boolean);

begin

CanClose := false;

end;

procedure TForm1.FormCreate(Sender: TObject);

var

ndoc,d,m,y: string;

begin

ini := Tinifile.Create(ExtractFilePath(ParamStr(0)) + 'VCM.ini');

EditInFolder.Text := ini.ReadString('FOLDERS', 'Input', '');

EditOutFolder.Text := ini.ReadString('FOLDERS', 'Output', '');

EditProcess.Text := ini.ReadString('CAMERA', 'Process', '');

Edit3.Text:= ini.ReadString('CAMERA', 'RunApp', '');

CheckBoxForRun.Checked := ini.ReadBool('CAMERA', 'RunProcess', true);

Label1.caption := ini.ReadString('ORG', 'Num', '00');

ini.free;

MakeFolders;

DateTimePicker1.Date := Date;

d:= InttoStr(DayOfTheMonth(Date));

if length(d)<2 then d:='0'+d;

m:= InttoStr(MonthOfTheYear(Date));

if length(m)<2 then m:='0'+m;

Edit2.Text := d + m + InttoStr(YearOf(Date) - 2000);

Form1.Height := 41;

// формируем новый номер документа

FDC.params.Database := ExtractFilePath(Application.ExeName) + 'VCM.SQLITE';

FDC.Open();

Query1.SQL.Add(' CREATE TABLE if not exists doc ( ');

Query1.SQL.Add(' id INTEGER PRIMARY KEY AUTOINCREMENT, ');

Query1.SQL.Add(' org INTEGER, ndoc INTEGER, fdoc TEXT (50), ');

Query1.SQL.Add

(' nfiles INTEGER, dt TEXT (30),tm TEXT(30), mon INTEGER,year INTEGER)');

Query1.ExecSQL;

Query1.SQL.Clear;

DateTimePicker1.Date := Date;

y:= InttoStr(YearOf(Date) - 2000);

if length(d)<2 then d:='0'+d;

m:= InttoStr(MonthOfTheYear(Date));

if length(m)<2 then m:='0'+m;

Query1.Open('select max(ndoc)+1 n from doc where mon='+m+' and year='+y);

ndoc := Query1.FieldByName('n').Asstring;

if ndoc='' then ndoc:='1';

while length(ndoc) < 5 do

ndoc := '0' + ndoc;

Edit1.Text := ndoc;

FDC.Close;

Timer1.Enabled := true;

end;

end.