Лабораторная работа №5

Баштового Олега

353м группы

unit MAIN;

interface

uses Winapi.Windows, System.SysUtils, System.Classes, Vcl.Graphics, Vcl.Forms,

Vcl.Controls, Vcl.Menus, Vcl.StdCtrls, Vcl.Dialogs, Vcl.Buttons, Winapi.Messages,

Vcl.ExtCtrls, Vcl.ComCtrls, Vcl.StdActns, Vcl.ActnList, Vcl.ToolWin,

Vcl.ImgList, System.Actions, Vcl.Imaging.pngimage,System.StrUtils, Vcl.Imaging.jpeg, Vcl.FileCtrl, RichEdit;

type

TMainForm = class(TForm)

MainMenu1: TMainMenu;

File1: TMenuItem;

FileNewItem: TMenuItem;

FileOpenItem: TMenuItem;

FileCloseItem: TMenuItem;

Window1: TMenuItem;

Help1: TMenuItem;

N1: TMenuItem;

FileExitItem: TMenuItem;

WindowCascadeItem: TMenuItem;

WindowTileItem: TMenuItem;

WindowArrangeItem: TMenuItem;

HelpAboutItem: TMenuItem;

OpenDialog: TOpenDialog;

FileSaveItem: TMenuItem;

FileSaveAsItem: TMenuItem;

Edit1: TMenuItem;

CutItem: TMenuItem;

CopyItem: TMenuItem;

PasteItem: TMenuItem;

WindowMinimizeItem: TMenuItem;

StatusBar: TStatusBar;

ActionList1: TActionList;

EditCut1: TEditCut;

EditCopy1: TEditCopy;

EditPaste1: TEditPaste;

FileNew1: TAction;

FileSave1: TAction;

FileExit1: TAction;

FileOpen1: TAction;

FileSaveAs1: TAction;

WindowCascade1: TWindowCascade;

WindowTileHorizontal1: TWindowTileHorizontal;

WindowArrangeAll1: TWindowArrange;

WindowMinimizeAll1: TWindowMinimizeAll;

HelpAbout1: TAction;

FileClose1: TWindowClose;

WindowTileVertical1: TWindowTileVertical;

WindowTileItem2: TMenuItem;

ToolBar2: TToolBar;

ToolButton1: TToolButton;

ToolButton2: TToolButton;

ToolButton3: TToolButton;

ToolButton4: TToolButton;

ToolButton5: TToolButton;

ToolButton6: TToolButton;

ToolButton9: TToolButton;

ToolButton7: TToolButton;

ToolButton8: TToolButton;

ToolButton10: TToolButton;

ToolButton11: TToolButton;

ImageList1: TImageList;

SaveDialog1: TSaveDialog;

Convertto1: TMenuItem;

procedure FileNew1Execute(Sender: TObject);

procedure FileOpen1Execute(Sender: TObject);

procedure HelpAbout1Execute(Sender: TObject);

procedure FileExit1Execute(Sender: TObject);

procedure BMP2JPEG(const SourceFileName: TFileName; DestFileName: TFileName);

procedure BMP2WMF(const SourceFileName: TFileName; DestFileName: TFileName);

procedure BMP2RTF(const SourceFileName :TFileName; DestFileName: TFileName);

procedure WMF2BMP (const SourceFileName: TFileName; DestFileName: TFileName);

procedure JPEG2BMP(const SourceFileName: TFileName; DestFileName: TFileName);

procedure BMP2EMF(const SourceFileName: TFileName; DestFileName: TFileName);

procedure BMP2ICO(const SourceFileName: TFileName; DestFileName: TFileName);

procedure ICO2BMP(const SourceFileName: TFileName; DestFileName: TFileName);

procedure PNG2BMP(const SOurceFileName: TFileName; DestFileName: TFileName);

procedure BMP2PNG(const SOurceFileName: TFileName; DestFileName: TFileName);

function FileExt(Fname: String): String;

procedure Convertto1Click(Sender: TObject);

procedure FormCreate(Sender: TObject);

private

{ Private declarations }

procedure CreateMDIChild(const FName: string);

public

{ Public declarations }

end;

var

MainForm: TMainForm;

ChosenDirectory: String;

implementation

{$R \*.dfm}

uses CHILDWIN, About;

procedure TMainForm.BMP2JPEG(const SourceFileName: TFileName; DestFileName: TFileName);

var jpgImg: TJPEGImage;

Bitmap: TBitmap;

FilePath: String;

begin

Bitmap := TBitmap.Create;

try

Bitmap.LoadFromFile(SourceFileName);

jpgImg := TJPEGImage.Create;

try

jpgImg.CompressionQuality := 100;

jpgImg.Assign(Bitmap);

jpgImg.Compress;

jpgImg.SaveToFile(DestFileName);

finally

jpgImg.Free;

end;

finally

Bitmap.Free;

end;

end;

procedure TMainForm.ICO2BMP(const SourceFileName: TFileName; DestFileName: TFileName);

var

winDC, destdc, srcDC: HDC;

oldBitmap: HBitmap;

iinfo: TICONINFO;

newBitMap: TBitmap;

Icon : TIcon;

begin

Icon := TIcon.Create;

newBitmap := TBitmap.Create;

Icon.LoadFromFile(SourceFileName);

newBitmap.Width := Icon.Width;

newBitmap.Height := Icon.Height;

newBitmap.Canvas.Draw(0, 0, Icon );

newBitmap.SaveToFile(DestFileName);

Icon.Free;

newBitmap.Free;

{newBitMap:=TBitMap.Create;

Icon := TIcon.Create;

Icon.LoadFromFile(SourceFileName);

GetIconInfo(Icon.Handle, iinfo);

WinDC := getDC(handle);

srcDC := CreateCompatibleDC(WinDC);

destDC := CreateCompatibleDC(WinDC);

oldBitmap := SelectObject(destDC, iinfo.hbmColor);

oldBitmap := SelectObject(srcDC, iinfo.hbmMask);

// Функция BitBlt выполняет передачу битовых блоков данных о цвете, соответствующих прямоугольнику пикселей из заданного исходного контекста устройства в целевой контекст устройства.

// SRCAND Комбинирует цвета исходных и целевых прямоугольников при помощи использования булева оператора И (AND). - черный

// SRCPAINT Комбинирует цвета источникового и целевого прямоугольников при помощи использования булева оператора ИЛИ (OR). - норм

BitBlt(destdc, 0 , 0 , Icon.width, Icon.height, srcDC, 0 , 0 , SRCPAINT);

NewBitmap.handle := SelectObject(destDC, oldBitmap);

DeleteDC(destDC);

DeleteDC(srcDC);

DeleteDC(WinDC);

newBitmap.SaveToFile(DestFileName);

newBitmap.Free; }

end;

procedure TMainForm.WMF2BMP(const SourceFileName: TFileName; DestFileName: TFileName);

var MetaFile : TMetafile;

Bitmap : TBitmap;

begin

Metafile := TMetaFile.Create;

Bitmap := TBitmap.Create;

try

MetaFile.LoadFromFile(SourceFileName);

Bitmap.Height := Metafile.Height;

Bitmap.Width := Metafile.Width;

Bitmap.Canvas.Draw(0, 0, MetaFile);

Bitmap.SaveToFile(DestFileName);

finally

Bitmap.Free;

MetaFile.Free;

end;

end;

procedure TMainForm.BMP2EMF(const SourceFileName: TFileName; DestFileName: TFileName);

var Metafile: TMetafile;

MetaCanvas: TMetafileCanvas;

Bitmap: TBitmap;

begin

Metafile := TMetaFile.Create;

MetaFile.Enhanced := true;

try

Bitmap := TBitmap.Create;

try

Bitmap.LoadFromFile(SourceFileName);

Metafile.Height := Bitmap.Height;

Metafile.Width := Bitmap.Width;

MetaCanvas := TMetafileCanvas.Create(Metafile, 0);

try

MetaCanvas.Draw(0, 0, Bitmap);

finally

MetaCanvas.Free;

end;

finally

Bitmap.Free;

end;

Metafile.SaveToFile(DestFileName);

finally

Metafile.Free;

end;

end;

procedure TMainForm.BMP2WMF(const SourceFileName: TFileName; DestFileName: TFileName);

var MetaFile : TMetafile;

Bitmap : TBitmap;

MetaCanvas: TMetafileCanvas;

begin

Metafile := TMetaFile.Create;

MetaFile.Enhanced := false;

try

Bitmap := TBitmap.Create;

try

Bitmap.LoadFromFile(SourceFileName);

Metafile.Height := Bitmap.Height;

Metafile.Width := Bitmap.Width;

MetaCanvas := TMetafileCanvas.Create(Metafile, 0);

try

MetaCanvas.Draw(0, 0, Bitmap);

finally

MetaCanvas.Free;

end;

finally

Bitmap.Free;

end;

Metafile.SaveToFile(DestFileName);

finally

Metafile.Free;

end;

end;

procedure TMainForm.BMP2ICO(const SourceFileName: TFileName; DestFileName: TFileName);

var IconInfo : TIconInfo;

Icon : TIcon;

BMP: TBitMap;

BMP2: TBitmap;

begin

BMP:=TBitmap.Create;

BMP.LoadFromFile(Sourcefilename);

BMP2:=TBitmap.Create;

BMP2.width:=32;

BMP2.height :=32;

// Функция SetStretchBltMode устанавливает режим растяжения точечного рисунка в заданном контексте устройства.

// BLACKONWHITE Выполняет булеву операцию И (AND), используя коды цвета для ликвидируемых и существующих пикселей. Если точечный рисунок - одноцветный битовый образ, этот режим сохраняет черные пиксели за счет белых пикселей.

// WHITEONBLACK - Выполняет булеву операцию ИЛИ (OR), используя коды цвета ликвидируемых и существующих пикселей. Если точечный рисунок - одноцветный битовый образ, этот режим сохраняет белые пиксели за счет черных пикселей.

// HALFTONE - Преобразует пиксели исходного прямоугольника в блоки пикселей в целевом прямоугольнике. Среднее значение цвета всего целевого блока пикселей подбирается близким по значению к цвету исходных пикселей.

SetStretchBltMode(bmp.Canvas.Handle, HALFTONE);// HALFTONE);

// Функция StretchBlt копирует точечный рисунок из исходного прямоугольника в целевой прямоугольник, растягивая или сжимая его, чтобы, в случае необходимости, подогнать под размеры целевого прямоугольника. Система растягивает или сжимает точечный рисунок согласно режиму растяжения, который в текущий момент установлен в приемном контексте устройства.

StretchBlt(bmp2.Canvas.Handle, 0, 0, bmp2.Width, bmp2.Height, bmp.Canvas.Handle, 0, 0, bmp.Width, bmp.Height, SRCCOPY);

Icon := TIcon.Create;

Icon.Width:=bmp2.Width;

Icon.Height:=bmp2.Height;

IconInfo.fIcon := true;

IconInfo.xHotspot := 0 ;

IconInfo.yHotspot := 0 ;

IconInfo.hbmMask := bmp2.Handle;

IconInfo.hbmColor := bmp2.Handle;

Icon.Handle := CreateIconIndirect(IconInfo);

Icon.SaveToFile(DestFileName);

Icon.Free;

BMP.Free;

BMP2.Free;

end;

procedure TMainForm.BMP2RTF(const SourceFileName :TFileName; DestFileName: TFileName);

var bi, bb, rtf: string ;

bis, bbs: Cardinal;

achar: String;

hexpict: string;

I: Integer;

Bitmap: TBitmap;

SS: TStringStream;

f:TextFile;

begin

Bitmap:=TBitmap.Create;

try

Bitmap.LoadFromFile(SourceFileName);

GetDIBSizes(bitmap.Handle, bis, bbs);

SetLength(bi, bis);

SetLength(bb, bbs);

GetDIB(bitmap.Handle, bitmap.Palette, PChar(bi)^, PChar(bb)^);

rtf := '{\rtf1 {\pict\dibitmap0 ';

SetLength(hexpict, (Length(bb) + Length(bi)) \* 2 );

I := 2 ;

for bis := 1 to Length(bi) do

begin

achar := IntToHex(Integer(bi[bis]), 2 );

hexpict[I - 1] := achar[ 1 ];

hexpict[I] := achar[ 2 ];

Inc(I, 2 );

end ;

for bbs := 1 to Length(bb) do

begin

achar := IntToHex(Integer(bb[bbs]), 2 );

hexpict[I - 1] := achar[ 1 ];

hexpict[I] := achar[ 2 ];

Inc(I, 2);

end ;

rtf := rtf + hexpict + ' }}';

AssignFile(f, DestFileName);

Rewrite(f);

Write(F,rtf);

CloseFile(f);

finally

Bitmap.Free;

end;

end;

procedure TMainForm.PNG2BMP(const SOurceFileName: TFileName; DestFileName: TFileName);

var png: TPngImage;

bmp: TBitMap;

begin

png:=TPNGImage.create;

try

png.LoadFromFile(SourceFileName);

bmp := TBitMap.Create;

try

bmp.Assign(png);

bmp.SaveToFile(DestFileName);

finally

bmp.free

end;

finally

png.free

end;

end;

procedure TMainForm.BMP2PNG(const SOurceFileName: TFileName; DestFileName: TFileName);

var png: TPNGImage;

bmp: TBitMap;

begin

bmp := TBitMap.Create;

try

bmp.loadfromfile(SourceFileName);

png := TPNGImage.Create;

try

png.Assign(bmp);

png.SaveToFile(DestFileName);

finally

png.Free

end;

finally

bmp.Free

end;

end;

procedure TMainForm.Convertto1Click(Sender: TObject);

var DestFName: String;

DestExt: String;

Ext: String;

begin

Ext:=FileExt(MainForm.ActiveMDIChild.Caption);

if SaveDialog1.Execute then

begin

DestFName := SaveDialog1.FileName;

DestExt:=ExtractFileExt(DestFName);

// анализируем выбранное расширение

case IndexText(DestExt, ['.bmp', '.emf', '.jpg', '.wmf', '.ico', '.png', '.rtf']) of

0: begin

// анализируем из какого расширения конвертировать в BMP

case IndexText(Ext, ['.emf', '.jpg', '.wmf', '.ico', '.png']) of

0: WMF2BMP(MainForm.ActiveMDIChild.Caption, DestFName);

1: JPEG2BMP(MainForm.ActiveMDIChild.Caption, DestFName);

2: WMF2BMP(MainForm.ActiveMDIChild.Caption, DestFName);

3: ICO2BMP(MainForm.ActiveMDIChild.Caption, DestFName);

4: PNG2BMP(MainForm.ActiveMDIChild.Caption, DestFName);

end;

end;

1: begin

BMP2EMF(MainForm.ActiveMDIChild.Caption, DestFName);

end;

2: begin

BMP2JPEG(MainForm.ActiveMDIChild.Caption, DestFName);

end;

3: begin

BMP2WMF(MainForm.ActiveMDIChild.Caption, DestFName);

end;

4: begin

BMP2ICO(MainForm.ActiveMDIChild.Caption, DestFName);

end;

5: begin

BMP2PNG(MainForm.ActiveMDIChild.Caption, DestFName);

end;

6: begin

BMP2RTF(MainForm.ActiveMDIChild.Caption, DestFName);

end;

end;

end;

end;

procedure TMainForm.CreateMDIChild(const FName: string);

var Child: TMDIChild;

begin

{ create a new MDI child window }

Child := TMDIChild.Create(Application);

//Child.Width:=405;

//Child.Height:=415;

Child.Caption := FName;

if FileExists(FName) then

begin

Child.Image1.Picture.LoadFromFile(FName);

end;

end;

procedure TMainForm.JPEG2BMP(const SourceFileName: TFileName; DestFileName: TFileName);

var jpeg: TJPEGImage;

bmp: TBitmap;

begin

jpeg := TJPEGImage.Create;

try

jpeg.LoadFromFile(SourceFileName);

bmp := TBitmap.Create;

try

bmp.Assign(jpeg);

bmp.SaveTofile(DestFileName);

finally

bmp.Free

end;

finally

jpeg.Free

end;

end;

procedure TMainForm.FileNew1Execute(Sender: TObject);

var

Child: TMDIChild;

begin

Child := TMDIChild.Create(Application);

Child.Width:=405;

Child.Height:=415;

Child.Caption := 'NONAME' + IntToStr(MDIChildCount + 1);

// CreateMDIChild('NONAME' + IntToStr(MDIChildCount + 1));

end;

procedure TMainForm.FileOpen1Execute(Sender: TObject);

var i: Integer;

begin

if OpenDialog.Execute then

begin

for I := 0 to OpenDialog.Files.Count - 1 do

begin

CreateMDIChild(OpenDialog.Files.Strings[i]);

Convertto1.Enabled:=true;

end;

end;

end;

procedure TMainForm.FormCreate(Sender: TObject);

begin

OpenDialog.Filter := 'All Files (\*.\*)|\*.\*|PNG Image File (\*.png)|\*.png|Bitmap (\*.bmp)|\*.bmp|Icons (\*.ico)|\*.ico|Metafiles (\*.wmf)|\*.wmf|Enhanced Metafiles (\*.emf)|\*.emf|JPEG Image File (\*.jpg)|\*.jpg';

end;

procedure TMainForm.HelpAbout1Execute(Sender: TObject);

begin

AboutBox.ShowModal;

end;

procedure TMainForm.FileExit1Execute(Sender: TObject);

begin

Close;

end;

function TMainForm.FileExt(Fname: String): String;

var Ext: String;

begin

Ext:=ExtractFileExt(FName);

case IndexText(Ext, ['.bmp', '.emf', '.jpg', '.wmf', '.ico', '.png']) of

0: begin

SaveDialog1.Filter := 'PNG Image File (\*.png)|\*.png|RTF document (\*.rtf)|\*.rtf|Icons (\*.ico)|\*.ico|Metafiles (\*.wmf)|\*.wmf|Enhanced Metafiles (\*.emf)|\*.emf|JPEG Image File (\*.jpg)|\*.jpg';

SaveDialog1.DefaultExt:='\*.png';

FileExt:= '.bmp';

end;

1: begin

SaveDialog1.Filter := 'Bitmap (\*.bmp)|\*.bmp';

SaveDialog1.DefaultExt:='\*.bmp';

FileExt:= '.emf';

end;

2: begin

SaveDialog1.Filter := 'Bitmap (\*.bmp)|\*.bmp';

SaveDialog1.DefaultExt:='\*.bmp';

FileExt:= '.jpg';

end;

3: begin

SaveDialog1.Filter := 'Bitmap (\*.bmp)|\*.bmp';

SaveDialog1.DefaultExt:='\*.bmp';

FileExt:= '.wmf';

end;

4: begin

SaveDialog1.Filter := 'Bitmap (\*.bmp)|\*.bmp';

SaveDialog1.DefaultExt:='\*.bmp';

FileExt:= '.ico';

end;

5: begin

SaveDialog1.Filter := 'Bitmap (\*.bmp)|\*.bmp';

SaveDialog1.DefaultExt:='\*.bmp';

FileExt:= '.png';

end;

end;

end;

end.

unit CHILDWIN;

interface

uses Winapi.Windows, System.Classes, Vcl.Graphics, Vcl.Forms, Vcl.Controls,

Vcl.StdCtrls, Vcl.ExtCtrls, Vcl.Menus, System.SysUtils, System.StrUtils;

type

TMDIChild = class(TForm)

Image1: TImage;

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

// procedure FileExt(Fname: String);

private

{ Private declarations }

public

{ Public declarations }

end;

implementation

{$R \*.dfm}

uses Main;

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

begin

Action := caFree;

end;

end.

unit about;

interface

uses Winapi.Windows, System.Classes, Vcl.Graphics, Vcl.Forms, Vcl.Controls,

Vcl.StdCtrls, Vcl.Buttons, Vcl.ExtCtrls;

type

TAboutBox = class(TForm)

Panel1: TPanel;

OKButton: TButton;

ProgramIcon: TImage;

ProductName: TLabel;

Version: TLabel;

Copyright: TLabel;

Comments: TLabel;

private

{ Private declarations }

public

{ Public declarations }

end;

var

AboutBox: TAboutBox;

implementation

{$R \*.dfm}

end.