**мундарижага қайтиш**

11-боб. **DELPHI НИНГ ГРАФИК ИМКОНИЯТЛАРИ**

[**Холст**](#м1)

[**Қалам ва чўтка**](#м2)

[**Матнларни чиқариш**](#м3)

[**Содда график элементларни чизиш учун методлар**](#м4)

**[Суратларни экранга чиқариш](%22%20%5Cl%20%22%D0%BC5)**

[**Битли тасвирлар**](#м6)

[**Мультипиликация**](#м7)

[**Базавий нуқта методи**](#м8)

[**Битли тасвирлардан фойдаланиш**](#м9)

[**Дастур ресурсидан битли тасвирларни юклаш**](#м10)

[**"Мультфильм" кўриш**](#м11)

Delphi тили дастурчиларга тасвирий воситалар ёрдамида экранга турли чизмалар, расмлар, иллюстрацияларн чиқаришга имкон беради.

Дастур графикани объектнинг (форма ёки Image компонентасининг) сиртига чиқаради. Объектнинг сиртига canvas ҳусусияти мос келади. Объект сиртига бирор график элементни (тўғри чизиқ, айлана, тўғри тўртбурчак ва х.к) чиқариш учун бу объектнинг canvas ҳусусиятига эҳтиёжга қараб метод қўллаш лозим. Масалан, Form1.Canvas.Rectangle (10,10,100,100) буйруғи дастур ойнасида тўғри тўртбурчак чизади.

**11.1. Холст**

Биз юқорида айтиб ўтганимиздек, дастур графикани чиқариши мумкин бўлган сиртга Canvas ҳусусияти мос келади. Ўз навбатида, canvas ҳусусияти — бу Tcanvas типидаги объект ҳисобланади. Бу типдаги методлар содда график элементларни (нуқта, чизиқлар, айлана, тўғри тўртбурчаклар ва х.к.) ясашга имкон беради. Ҳусусиятлари эса график элементларнинг характеристикаларини (ранг, қалинлиги, чизиқларнинг стиллари, соҳаларга фон бериш усуллари, матнларга шрифтлар ва х.к.) белгилаб беради.

**11.1-расм.** Холстнинг координаталари

Содда график элементларни чиқариш методлари Canvas ҳусусиятларини қандайдир абстракт холст деб қарайди ва унда чизиш амалларини бажариши мумкин. (canvas – "сирт", "расм чизиш учун холст" деб таржима қилинади.) Холст алоҳида нуқталар-пикеллардан ташкил топган. Пикселнинг ҳолати унинг горизонтал (X) ва вертикал (Y) координаталари билан белгиланади. Чап юқори пикселнинг координаталари (0, 0) га тенг. Координаталар юқоридан пастга қараб, чапдан ўнг томонга қараб ўсади. (11.1-расм). Холст ўнг қуйи нуқтасининг координатаси холстнинг ўлчамларига боғлиқ.

Холстнинг ўлчамларини расмлар (Image) соҳасининг Height ва width ҳусусиятлари ёки форманинг ҳусусиятлари бўлган ClientHeight ва Clientwidth лар ёрдамида аниқлаш мумкин.

менюга

**11.2. Қалам ва чўтка**

Рассом расмларни одатда қалам ва чўтка ёрдамида чизади. Холст сиртида содда графикани ясаш таъминлайдиган методлар ҳам қалам ва чўткадан фойдаланади. Қалам чизиқлар ва контурларни чизса, чўтка контурлар билан чегараланган соҳаларни бўяш учун қўлланади.

Қалам ва чўткага Реn (қалам) ва Brush (чўтка) ҳусусиятлари мос келади. Улар мос раавишда TPen ва Tbrush типидаги объектлар ҳисобланади. Бу объектларнинг қийматлари экрандаги график элементларнинг кўринишларини белгилаб беради.

**Қалам.** Қалам нуқта, чизиқлар ва геометрик фигураларни (тўғри тўртбурчак, айлана, эллипслар, ёйлар ва х.к.) чизиш учун ишлатилади. Холст сиртида қалам қолдираётган чизиқнинг кўриниши TPen объектининг ҳусусиятларига боғлиқ. Бу ҳусусиятлар 11.1-жадвалда келтирилган.

 **TPen объектининг ҳусусиятлари 11.1-жадвал**

|  |  |
| --- | --- |
| Ҳусусияти  | Мазмуни  |
| Color | Чизиқнинг ранги |
| Width | Чизиқнинг қалинлиги |
| Style | Чизиқнинг кўриниши |
| Mode | Акслантириш режими |

**Color** ҳусусияти қалам билан чизилаётган чизиқ рангини белгилайди. 11.2-жадвалда color ҳусусиятининг қиймати сифатида фойдаланиш мумкин бўлган номланган константалар (Tcolor типи) рўйхати келтирилган.

 **Color ҳусусиятининг қийматлари 11.2-жадвал**

|  |  |  |  |
| --- | --- | --- | --- |
| **Константа** | **Ранг** | **Константа** | **Ранг** |
| clBlack | Қора | clSilver | Кумуш ранг |
| clMaroon | Каштан ранг | clRed | қизил |
| clGreen | Яшил  | clLime | Салат ранг  |
| clOlive | Зайтун ранг | clBlue | Мовий |
| clNavy | Тўқ кўк | clFuchsia | Тўқ қизил  |
| clPurple | Қизил | clAqua | Феруза ранг |
| clTeal | Кўкимтир яшил | clWhite | Оқ  |
| clGray | Кул ранг |  |  |

**Width** ҳусусияти чизиқ қалинлигини (пикселларда) кўрсатади. Масалан:

Canvas. Pen. Width :=2

буйруғи қалинлиги 2 пиксел бўлган чизиқни англатади.

**Style** ҳусусияти чизиқнинг кўринишини (стилини) билдиради. Чизиқ узлуксиз ёки узлукли (узуқ-узуқ) кўринишларидан бирида бўлади. 11.3-жадвалда чизиқнинг стилини белгиловчи номланган константалар рўйхати келтирилган. Пунктир чизиғининг қалинлиги 1 пикселдан катта бўла олмайди. Агар Pen.width ҳусусиятининг қиймати 1 дан катта бўлса, у ҳолда пунктир чизиғи узлуксиз кўринишда экранга узатилади.

**Реn.width ҳусусиятининг қийматлари 11.3-жадвал**

|  |  |
| --- | --- |
| **Константа** | **Чизиқнинг кўриниши** |
| psSolid | Узлуксиз чизиқ |
| psDash | Узун штрихли пунктир чизиқ  |
| psDot | Калта штрихли пунктир чизиқ |
| psDashDot | Узун ва калта штрихли пунктир чизиқ |
| psDashDotDot | Битта узун, иккита кисқа штрихли пунктир чизиқ |
| psClear | Чизиқ кўринмайди. (соҳа чегараси кўринмаслиги керак бўлган ҳолларда қўлланади)  |

**Mode** ҳусусияти холстнинг нуқталарининг рангига боғлиқ равишда чизиқнинг нуқталарининг ранги қандай ҳосил қилинишини аниқлайди. Агар кўрсатилмаган бўлса. У ҳолда чизиқлар Pen.Color ҳусусиятининг қийматида кўрсатилган ранг билан чизилади. Аммо, дастурчи фон рангига нисбатан инверсион рангни ҳам белгилаши мумкин. Бундай ҳол фон рангига боғлиқ бўлмаган ҳолда чизиқ тўлалигича кўринишини таъминлайди. 11.4 –жадвалда Pen.Mode ҳусусиятининг айрим қийматлари келтирилган.

 **Pen.Mode ҳусусияти чизиқ рангига таъсир кўрсатади. 1.4 –жадвал**

|  |  |
| --- | --- |
| **Константа** | **Чизиқнинг ранги** |
| pmBlack | Қора, Pen. Color ҳусусияти қийматига боғлиқ эмас |
| pmWhite | Оқ, Pen. Color ҳусусияти қийматига боғлиқ эмас |
| pmCopy | Чизиқ ранги Pen. Color ҳусусияти қийматига кўра танланади.  |
| pmNotCopy | Чизиқ ранги Pen. Color ҳусусияти қийматига инверсион рангда бўлади.  |
| pmNot | Чизиқ нуқталарининг ранги чизиқ чизиладиган холст рангига нисбатан инверсион бўлади.  |

**Чўтка.** Методлар чўткани (canvas.Brush) ёпиқ соҳаларни (масалан, геометрик фигураларни) чизиш ва бўяш учун ишлатади. Чўтка, объект сифатида иккита ҳусусиятга эга. Улар 11.5-жадвалда берилган.

**TBrush (чўтка) ҳусусиятининг қийматлари 11.5-жадвал**

|  |  |
| --- | --- |
| Ҳусусияти  | маъноси |
| Color | Ёпиқ соҳани бўяйди  |
| Style | Соҳага фон бериш стили (типи)  |

Контурнинг ёпиқ ички соҳаси бўялган ёки фон берилган бўлиши мумкин. 1-ҳолда соҳа тўлалигича фонни бекитади, 2-чисида эса штрихланмаган жойларда фон кўриниб туради. Color ҳусусиятининг қиймати сифатида TColor типидаги ихтиёрий номланган константадан фойдаланиш мумкин. (11.2-жадвалга қаранг)

Соҳага фон бериш стилини белгиловчи номланган константалар 11.6-жадвалда берилган.

 **Brush.style ҳусусиятининг қийматлари 11.6-жадвал**

|  |  |
| --- | --- |
| **Константа** | **Соҳани тўлдириш типи (фони)**  |
| bsSolid | Узлуксиз фон билан тўлдириш |
| bsClear | Соҳа бўялмайди |
| bsHorizontal | Соҳага горизонтал штрихли фон бериш |
| bsVertical | Соҳага вертикал штрихли фон бериш |
| bsFDiagonal | Соҳага олд томонга қийшайган штрихли фон бериш |
| bsBDiagonal | Соҳага орқага қийшайган штрихли фон бериш |
| bsCross | Катаксимон штриховкали фон  |
| bsDiagCross | Қийшиқ чизиқли катаксимон фон  |

Намуна сифатида 11.1-листингда **соҳага фон бериш стиллари** дастури келтирилган. Унинг ёрдамида экранга (11.2-расм) қора рангли ва турли стиллар билан фон берилган саккизта тўғри тўртбурчак тасвири чиқарилади.

**11.2-расм. С**оҳага фон бериш стиллари дастурининг ойнаси

**11.1-листинг. Фон бериш стиллари.**

unit ctil\_;

interface

uses Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms, Dialogs, ExtCtrls;

type

 TForm1 = class(TForm)

 Image1: TImage;

 procedure Image1Click(Sender: TObject);

 private

 { Private declarations }

 public

 { Public declarations }

 end;

var

 Form1: TForm1;

implementation

{$R \*.dfm}

procedure TForm1.Image1Click(Sender: TObject);

const

bsName: array[1..8] of string = ('bsSolid','bsClear','bsHorizontal',

'bsVertical','bsFDiagonal','bsBDiagonal', 'bsCross','bsDiagCross');

var

x,y: integer; ⁄⁄ тўртбурчак юқори чап бурчагининг координаталари

w,h: integer; ⁄⁄ тўғри тўртбурчакнингг кенглиги ва баландлиги

bs: TBrushStyle; ⁄⁄ фон бериш стиллари

k: integer; ⁄⁄ фон бериш стили номери

i,j: integer;

begin

w := 40; h := 40; ⁄⁄ Тўғри тўртбурчакнинг ўлчамлари

y := 20;

for i := 1 to 2 do

begin

x := 10;

for j := 1 to 4 do

begin

k := j + (i-1)\*4; ⁄⁄ фон бериш стили номери

case k of

1: bs := bsSolid;

2: bs := bsClear;

3: bs := bsHorizontal;

4: bs :­= bsVertical;

5: bs := bsFDiagonal;

6: bs := bsBDiagonal;

7: bs := bsCross;

8: bs := bsDiagCross; end;

⁄⁄ тўғри тўртбурчакнинг чиқариш

Canvas.Brush.Color := clGreen;

⁄⁄ бўяш ранги - яшил

Canvas.Brush.Style := bs;

⁄⁄ фон бериш стили

Canvas . Rectangle (x, y, x+w, y+h) ;

⁄⁄ стил номерини чиқариш

Canvas.Brush.Style := bsClear;

Canvas.TextOut(x, y-15, bsName[k]);

x := x + w + 30;

end;

y := y + h + 30;

end;

end;

end.

менюга

**11.3. Матнларни чиқариш**

Матнларни график объектнинг сиртига чиқариш учун TextOut методидан фойдаланилади. Бу методни умумий кўринишда қўйидаги буйруғи билан чақирилши мумкин:

***Объект.Canvas.TextOut(x, у, Матн)***

Бу ерда ***объект*** — сиртига матн чиқариладиган объектнинг номи; ***х, у*** – график сиртдаги нуқтанинг координаталари (11.3-расм); ***Матн***-белгили типдаги ўзгарувчи ёки константа, унинг қиймати матн сифатида сиртнинг ***(x, y)*** координатали нуқтасидан бошлаб экранга чиқарилади.

11.3-расм. Матн чиқариладиган соҳанинг координаталари

Матнларни экранга чиқариш учун фойдаланиладиган шрифт Сanvas объектининг Font ҳусусияти билан белгиланади. 11.7-жадвалда Tfont типидаги объектнинг ҳусусиятлари келтирилган. Улардан TextOut ва TextRect методларида фойдаланиш мумкин.

 TFont объектининг ҳусусиятлари 11.7-жадвал

|  |  |
| --- | --- |
| Щусусияти  | Мазмуни |
| Name | Фойдаланиладиган шрифт номи. +иймат сифатида шрифт номи ёзилади. Масалан, Arial . |
| Size  | Шрифтнинг пунктлардаги ылчами. Пункт – полиграфияда =ылланадиган шрифт ылчамларининг ылчов бирлиги. 1 пункт – 1 ⁄ 72 дюймга тенг .  |
| Style  | Харфларни ёзиш усули. Оддий, =орайтирилган, курсив, тагига чизилган, устидан чизилган былиши мумкин. Бу стилларни =уйидаги константалар ёрдамида берилади: fsBold (=орайтирилган), fsItalic (курсив), fsUnderline (тагига чизилган), fsStrikeout (устидан чизилган).Style щусусияти тыплам щисобланади, шунинг учун эхтиёжга =араб мураккаб комбинацияли шрифтларни танлаш мумкин. Масалан, "=орайтирилган курсив" шрифти Объект. Canvas . Font := [fsBold, fsItalic]кыринишида ёзилади.  |
| Color  | Белгиларнинг ранги. +иймат сифатида Tcolor типидаги константалардан фойдаланиш мумкин. |

**Диққат!:** Матнларнинг чиқариш соҳаси чўтканинг жорий ранги билан бўялган бўлади. Шунинг учун матнни экранга чиқаришдан аввал Brush.Color ҳусусиятига bsClear қийматини бериш ёки чўткага матн чиқариладиган сиртнинг ранги билан бир ҳил қийматни ўзлаштириш лозим.

Қуйидаги дастур парчаси форма сиртига матнларни чиқариш учун Textout функциясидан фойдаланишга намуна бўла олади.

**with** Form1.Canvas **do begin**

⁄⁄ шрифтнинг характеристикаларини белгилаш

Font.Name := 'Tahoma';

Font.Size := 20;

Font.Style := [fsltalic, fsBold] ;

Brush.Style := bsClear; ⁄⁄ матнни чиқариш соҳаси бўялмайди

TextOut(0, 10, 'Borland Delphi 7');

**end;**

Матнни Textout методи билан сиртга чиқарилганидан сўнг, чиқариш кўрсаткичи (қалам) матн чиқарилган соҳанинг ўнг юқори бурчагига боради.

Айрим ҳолларда дастурни ишилаб чиқиш вақтида узунлиги номаълум бўлган ахборотдан кейин қандайдир матнни чиқаришга тўғри келади. Масалан, бу бирор сонни харфлар орқали ёзувидан кейин чиқарилиши талаб қилинган "сўм" сўзи бўлсин. Бу ҳолда чиқарилган матннинг ўнг томондаги координатасини билиш лозим бўлади. TextOut методи билан чиқарилган матннинг ўнг томондаги координасини PenPos ҳусусияти ёрдамида аниқлаш мумкин. Қуйидаги дастур парчаси матннинг сатрларини иккита TextOut ёрдамида чиқаришни намойиш қилади.

 **with** Form1.Canvas **do begin**

TextOut(0, 10, 'Borland ') ;

TextOut(PenPos.X, PenPos.Y, 'Delphi 7');

**end;**

менюга

**11.4. Содда график элементларни чизиш учун методлар**

Ихтиёрий расм, чизма ва схемаларни содда график элементларнинг (нуқта, чизиқ, айлана, ёй ва х.к.) тўпламидан иборат деб қараш мумкин. Шунинг учун экранда керакли тасвирни ҳосил қилиш учун дастур бу тасвирни ташкил қилувчи график содда элементларни чизишни (экранга чиқаришни) таъминлаши лозим.

Содда график элементларни компонента сиртида (форма ёки илюстрацияларни чиқариш соҳасида) чизиш масаласи шу компонентанинг Canvas ҳусусиятининг эхтиёжга қараб, мос методларини қўллаш орқали ҳал қилиниши мумкин.

**Чизиқлар.** Тўғри чизиқларни чизиш учун LinеТо методидан фойдаланилади. Бу методни умумий кўринишда қуйидаги усулда чақирилади:

***Компонент.Canvas.LineTo(x,у)***

***LinеТо*** методи қаламнинг жорий позициясидан бошлаб, методга мурожаат қилинганда кўрсатилган координатали нуқтагача бўлган тўғри чизиқни (кесмани) ясайди. Чизиқнинг бошланғич нуқтасини қаламни график сиртнинг керакли позициясига суриш усули билан кўрсатиш мумкин. Бу вазифани ***MoveTo*** методи бажаради. Унинг параметрлари иккита бўлиб, қаламнинг навбатдаги позициясини кўрсатувчи нуқтани англатади. Чизиқнинг кўриниши, (ранги, қалинлиги ва стили) чизиқ чизилиши талаб қилинган график сиртнинг Pen ҳусусиятининг қийматлари билан аниқланади.

Кўпинча, ҳисоблашлар натижаларини графиклар кўринишида ифодалаш қулай ҳисобланади. Тушуниш осон бўлиши ҳамда кўргазмалиликни ошириш учун графикларни координаталар ўқи ва рақамли тўр фонидан фойдаланган ҳолда ясалади. 11.2-листингда форма сиртига координаталар ўқи ва рақамли тўрни чиқарувчи дастурнинг матни келтирилмоқда. (11.4-расм)

**11.4-расм.**  Координата тўри дастурининг ойнаси

**11.2-листинг. Координата ўқлари ва рақамланган тўр**

unit grid\_;

interface

uses Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs, StdCtrls;

type

 TForm1 = class(TForm)

 procedure FormPaint(Sender: TObject);

 private

 { Private declarations }

 public

 { Public declarations }

 end;

var

 Form1: TForm1;

implementation

{$R \*.DFM}

procedure TForm1.FormPaint(Sender: TObject);

var

 x0,y0:integer; ⁄⁄ координата ўқларининг бошланғич нуқтаси

 dx,dy:integer; ⁄⁄ координата тўрининг қадами (пикселларда)

 h,w:integer; ⁄⁄ координата тўрининг чиқариш соҳасининг баландлиги ва кенглиги

 x,y:integer;

 lx,ly:real; ⁄⁄ тўр чизиқларини рақамлаш (X ва Y бўйича)

 dlx,dly:real; ⁄⁄ тўр чизиқлариниг белгилаш қадами (X ва Y бўйича)

 cross:integer; ⁄⁄ рақамланмаган тўр чизиқларининг ҳисоблагич

 dcross:integer; ⁄⁄ рақамланганлари орасида рақамланмаган чизиқларнинг миқдори

begin

 x0 := 30; y0 := 220; ⁄⁄ ўқлар (40,250) нуқтадан бошланади.

 dx := 40; dy := 40; ⁄⁄ координата тўрининг қадами 40 пиксел

 dcross := 1; ⁄⁄ Х тўрнинг чизиқларини белгилаш: 1 - ҳар бирини

 ⁄⁄ 2 - биттадан оралатиб

 ⁄⁄ 3 - иккитадан оралатиб

 dlx := 0.5; ⁄⁄ Х ўқининг белгилаш қадами

 dly := 1.0; ⁄⁄ У ўқининг белгилаш қадами: 1, 2, 3 ва х.к.

 h := 200;

 w := 300;

 with form1.Canvas do

 begin

 cross := dcross;

 MoveTo(x0,y0); LineTo(x0 , y0-h); ⁄⁄ X ўқи

 MoveTo(x0, y0); LineTo(x0 + w, y0); ⁄⁄ Y ўқи

 ⁄⁄ Х ўқи бўйича бўлаклаш, тўр ва рақамлаш

 x := x0 + dx;

 lx := dlx;

 repeat

 MoveTo(x,y0-3); LineTo(x,y0 + 3); ⁄⁄ бўлаклаш

 cross := cross-1;

 if cross = 0 then ⁄⁄ рақамлаш

 begin

 TextOut(x-8, y0 + 5, FloatToStr(lx));

 cross := dcross;

 end;

 Pen.Style := psDot;

 MoveTo(x, y0-3); LineTo(x, y0-h); ⁄⁄ тўрнинг чизиқлари

 Pen.Style := psSolid;

 lx := lx + dlx;

 x := x + dx;

 until (x>x0 + w);

 ⁄⁄ Y ўқи бўйича бўлаклаш, тўр ва рақамлаш

 y := y0-dy;

 ly := dly;

 repeat

 MoveTo(x0-3, y); LineTo(x0 + 3,y); ⁄⁄ бўлаклаш

 TextOut(x0-20, y, FloatToStr(ly)); ⁄⁄ рақамлаш

 Pen.Style := psDot;

 MoveTo(x0 + 3, y); LineTo(x0 + w, y); ⁄⁄ тўрнинг чизиқлари

 Pen.Style := psSolid;

 y := y-dy;

 ly := ly + dly;

 until (y<y0-h);

 end;

end;

end.

**Дастурни ишга тушириш**

Келтирилган дастурнинг ўзига ҳос томони шундаки, у тўрнинг қадами ва рақамлаш усулини белгилашга имкон беради. Бундан ташқари, дастур Х ўқининг тўрининг ҳар бир чизиқларини эмас, балки оралатиб ҳам рақамлай олади. Бунинг сабаби шуки, агар рақамлашда қатнашадиган сонлар бир нечта рақамдан иборат бўлса, рақамлашда қатнашадиган сонларни экранга чиқарганда уларнинг айрим рақамлари бир-бирининг устига тушиб қолиши эҳтимоли олдини олишдан иборат.

**Синиқ чизиқ. *Polyline*** методи синиқ чизиқ чизиш учун хизмат қилади. Бу метод параметр сифатида Tpoint типидаги массивни қабул қилади. Массивнинг ҳар бир элементи ***x*** ва ***y*** майдонли ёзувдан иборат бўлиб, синиқ чизиқнинг синиқ нуқтасини белгилайди. Polyline методи координатаси массивда жойлашган синиқ чизиқнинг синиш нуқталарини кетма-кет кесмалар билан бирлаштиради: биринчини иккинчи билан, иккинчини учинчи билан ва х.к.

Polyline методидан ёпиқ контурларни чизиш учун ҳам фойдаланиш мумкин. Бунинг учун методнинг параметри сифатида иштирок этаётган массивнинг биринчи ва охирги элементлари бир ҳил бўлиши лозим. Намуна сифатида 11.3-листингда берилган дастурни кўрайлик. Бу дастур диалог ойнаси сиртида сичқонча тугмаси чертилган жойда беш қиррали юлдуз тасвирини (11.5-расм) ясайди. Юлдуз чизилган ранг сичқончанинг қайси тугмаси чертилишига боғлиқ. Сичқонча тугмасини чертиш (MouseDown ходисаси) учун ходисаларни қайта ишлаш процедураси юлдуз чизиш процедураси StarLine ни чақиради ва унга параметр сифатида сичқонча нуқтаси чертилган нуқтанинг координаларини узатади. Юлдузни StarLine процедураси чизади. У параметр сифатида юлдуз марказининг координаталарини ҳамда ҳолстни олади. Дастлаб, юлдузнинг учлари ва чуқурликларининг координаталари ҳисобланади ва бу маълумотлар p массивга ёзилади. Сўнгра бу массив PolyLine методига параметр қилиб узатилади. P массив элементларини ҳисоблашда қутб координаталар системасидан фойдаланамиз. Бунинг учун бурчаклар радианларда ифодаланган бўлиши керак.

**11.3-листинг. Ёпиқ контур (юлдуз) чизиш**

unit Stars\_;

interface

uses Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms, Dialogs, StdCtrls;

type

 TForm1 = class(TForm)

 procedure FormMouseDown(Sender: TObject; Button: TMouseButton;

 Shift: TShiftState; X, Y: Integer);

 private

 { Private declarations }

 public

 { Public declarations }

 end;

var

 Form1: TForm1;

implementation

{$R \*.dfm}

procedure StarLine(x0,y0,r: integer; Canvas: TCanvas);

 ⁄⁄ x0,y0 – Юлдуз марказининг координатаси

 ⁄⁄ r – юлдузнинг радиуси

var

 p : array[1..17] of TPoint; ⁄⁄ учлар чуқурликлар массиви

 a: real; ⁄⁄ Учлар учун бурчак

 i: integer;

begin

 a := 0; ⁄⁄ ўнг учдан ясашни бошлаш учун

 for i := 1 to 16 do

 begin

 if (i mod 2 = 0) then

 begin ⁄⁄ чуқурлик координаталари

 p[i].x := x0 + Round(r ⁄ 2\*cos(a\*2\*pi ⁄ 360));

 p[i].y := y0-Round(r ⁄ 2\*sin(a\*2\*pi ⁄ 360));

 end

 else

 begin ⁄⁄ учнинг координаталари

 p[i].x := x0 + Round(r\*cos(a\*2\*pi ⁄ 360));

 p[i].y := y0 - Round(r\*sin(a\*2\*pi ⁄ 360));

 end;

 a := a + 22.5;

 end;

 p[17].X := p[1].X; ⁄⁄ Юлдуз контурини ёпиш учун

 p[17].Y := p[1].Y;

 Canvas.Polyline(p); ⁄⁄ юлдузни чизиш

end;

⁄⁄ сичқонча тугмаларини чертилиши

procedure TForm1.FormMouseDown(Sender: TObject; Button: TMouseButton; Shift: TShiftState; X, Y: Integer);

begin

 if Button = mbLeft ⁄⁄ чап тугма чертилганми ?

 then Form1.Canvas.Pen.Color := clBlack

 else Form1.Canvas.Pen.Color := clRed;

 StarLine(x, y, 30,Form1.Canvas);

end;

end.

**Дастурни ишга тушириш**

**Эслатма:** P массивнинг ўлчами учлар ва чуқурликлар сонидан биттага кўп ҳамда массивнинг биринчи ва охирги элементлари устма-уст тушади.

11.5-расм. Юлдуз

**Айлана ва эллипс.** Эллипс ёки айлана тасвирини **ҳосил** қилиш учун ***Ellipse*** методидан фойдаланилади. Эллипс ёки айлана чизиш бу методнинг параметрларига боғлиқ. Методни ишга туширишнинг умумий кўриниши қуйидагича:

 ***Объект.Canvas.Ellipse(x1,y1, х2,у2]***

Бу ерда ***объект*** — сиртида расм чизиш талаб қилинган объектнинг (компонентанинг) номи; ***x1, y1, х2, у2*** — айлана ёки эллипсга ташқи чизилган тўғри тўртбурчакнинг координаталари. Демак, агар тўғри тўртбурчак квадратдан иборат бўлса айлана, акс ҳолда эллипс чизилади. (11.6-расм).

Эллипс чизиқларининг ранги, қалинлиги ва стили Pen ҳусусиятининг қийматлари билан аниқланади, эллипснинг ички ранги ва фон бериш стили расм чизиладиган сиртнинг (canvas нинг) Brush ҳусусияти қийматлари билан белгиланади.

11.6-расм. Чизиладиган фигура Ellipse методининг параметрларига боғлиқ

**Ёй.** Ёйларни ***Arc*** методи билан ясаш мумкин. Бу методни умумий кўринишда қуйидагича ёзиш мумкин:

 ***Объект.Canvas.Arc(x1,y1,х2,у2,х3,у3,х4,у4)***

Бу ерда ***x1, y1, х2, у2*** — чизилаётган ёйнинг асоси бўлган эллипснинг (айлананинг) параметрлари; ***х3, у3*** — ёйнинг бошланиш нуқтасини кўрсатувчи параметрлар; ***х4, у4*** —ёйнинг тугаш нуқтасини белгилайдиган параметрлар.

Бошланғич (охирги) нуқта – бу эллипс чегарасининг эллипс марказидан (x3, y3) ҳамда (x4, y4) координатали нуқталарга ўтказилган тўғри чизиқлар орасида ётган қисми. Ёйлар соат милларига тескари йўналишда ясалади. (11.7-расм).

Ёйнинг ранги, қалинлиги ва чизиқларининг стилини расм чизилаётган (canvas) сиртнинг Pen ҳусусияти қийматлари билан кўрсатилади.

11.7-расм. Ёй эллипс (айлана) нинг маълум бир қисмидан иборат.

**Тўғри тўртбурчак.**  Тўғри тўртбурчаклар ***Rectangle***  методи билан чизилади. Бу методнинг умумий кўриниши қуйидагича:

***Объект.Canvas.Rectangle(x1, y1,x2, y2)***

Бу ерда ***Объект*** — сиртида расм чизиладиган объект (компонент) нинг номи; ***x1, y1*** ва ***х2, у2*** — тўғри тўртбурчакнинг чап юқори ва ўнг қуйи бурчакларнинг координаталари.

***RoundRec*** методи учлари ёйсимон бўлган тўғри тўртбурчак ясаш учун фойдаланилади. Бу методни умумий кўринишда

***Объект.Canvas.RoundRec(x1,y1,х2, у2, х3, у3)***

тарзида ишга туширилади. Бу ерда ***x1, y1, х2, у2*** – ёйсимон учли тўғри тўртбурчакка ташқи чизилган тўғри тўртбурчакнинг координаталари; ***х3***, ***у3*** — чорак қисми тўғри тўртбурчак учи сифатида чизиладиган эллипснинг ўлчамлари (11.8-расм).

11.8-расм. RoundRec методи учлари ёйсимон тўғри тўртбурчак чизади

Тўғри тўртбурчак чизиқларининг ранги, қалинлиги ва стили Pen ҳусусиятининг қийматлари билан аниқланади. Тўғри тўртбурчакнинг ички ранги ва фон бериш стили расм чизиладиган сиртнинг (canvas нинг) Brush ҳусусияти қийматлари билан белгиланади.

Тўғри тўртбурчак ясаш учун яна иккита метод мавжуд. Бу методлар қалам ўрнига мўйқалам билан чизади. ***FillRect*** методи бўялган тўғри тўртбурчак, ***FrameRect*** методи эса фақат тўғри тўртбурчак контурини чизиш учун фойдаланилади. Бу методларнинг ҳар иккисида фақат битта параметр - TRect типидаги структура қатнашади. TRect структурасининг майдонлари тўғри тўртбурчакли соҳанинг координаталарини сақлайди, улар Rect функцияси ёрдамида аниқланиши мумкин.

**Кўпбурчак. *Polygon*** методи кўпбурчак чизиш учун хизмат қилади. Бу методнинг параметри Tpoint типдаги массивдан иборат. Бу масивнинг ҳар бир элементи ёзув бўлиб, иккита майдон ***х*** ва ***у*** ларни ўз ичига олади. Бу ёзув-нуқта кўпбурчакнинг битта учини ифодалайди. Polygon методи координаталари массивда кўрсатилган нуқталарни кетма-кет кесмалар билан бирлаштиради: биринчисини иккинчи билан, иккинчисини учинчиси билан ва х.к. Сўнгра охирги нуқта ва биринчи нуқталар туташтирилади.

Кўпбурчак чизиқларининг ранги, қалинлиги ва стили Pen ҳусусиятининг қийматлари билан аниқланади, кўпбурчакнинг ички ранги ва фон бериш стили расм чизиладиган сиртнинг (canvas нинг) Brush ҳусусияти қийматлари билан белгиланади, соҳа сўйқаламнинг жорий ранги ва стилида бўялади. Қуйидаги процедурада polygon методи билан учбурчак ясалади.

**procedure** TForm1.Button2Click(Sender: TObject);

**var**

pol: array[1..3] **of** TPoint; ⁄⁄ Учбурчак учларининг координаталари

**begin**

pol[1].x := 10; pol[1].y := 50;

pol[2].x := 40; pol[2].y := 10;

pol[3].х := 70; pol[3].у := 50;

Form1.Canvas.Polygon(pol);

**end;**

**Сектор.** Секторларни ***Pie*** методи ёрдамида чизиш мумкин. Бу методни умумий кўринишда қуйидаги бўйруқ билан чақирилади:

 ***Объект. Canvas.Pie(x1,y1,x2,y2,х3,у3,х4,у4)***

Бу ерда ***x1, y1, х2, у2*** — секторнинг асоси бўлган эллипсни (айланани) аниқлайдиган параметрлар; ***х3, у3, х4, у4 -*** секторнинг чегараси бўлган тўғри чизиқларнинг охирги нуқталарининг координаталари.

Тўғри чизиқларнинг бошланғич нуқталари эллипс (айлана) маркази билан устма-уст тушади. Сектор соат милларига тескари йўналишда кесиб олинади: координаталари (хЗ, у3) бўлган нуқтадан бошланиб, (х4, у4) координатали нуқтада тугайди. (11.9-расм).

**11.9-расм.**  Pie методининг параметрларига секторнинг боғлиқлиги

**Нуқта.** Дастур график чизадиган сиртга Canvas объекти мос келади. TColor типидаги икки ўлчовли массивдан иборат бўлган ***Pixels*** ҳусусияти график сиртнинг ҳар бир нуқтасининг ранги ҳақидаги маълумотни сақлайди. Pixels ҳусусиятидан фойдаланиб, график сиртнинг ихтиёрий нуқтасига талаб қилинган рангни бериш мумкин. Масалан,

Form1.Canvas.Pixels[10,10] := clRed;

буйруғи форма сиртидаги (10, 10) координатали нуқтага қизил ранг беради.

Pixels массивининг ўлчамлари график сиртнинг ўлчамлари билан белгиланади. Форманинг график сиртининг (клиентлар соҳаси деб ҳам аталади ишчи соҳанинг) ўлчамлари Clientwidth ва ClientHeight ҳусусиятларининг қийматлари билан аниқланади, Image компонентасининг ўлчамлари эса Width ва Height ҳусусиятларининг қийматлари билан белгиланади. Форманинг иши соҳасининг чап юқори бурчагига pixels[0,0], ўнг қуйи бурчакка эса -Pixels[Clientwidth-1,ClientHeight-1] элементлари мос келади.

Pixels ҳусусиятидан графиклар ясашда фойдаланилади. Одатда графиклар берилган формулалар бўйича ҳисоблашлар натижаси асосида қурилади. Функция аргументининг ўзгариш диапазони бошланғич маълумот деб қабул қилинади. Функция қийматларининг ўзгариш соҳаси ва диапазонини эса ҳисоблаб топиш мумкин. Олинган маълумотлар асосида масштабни топиш мумкин. Масштаб ясалаётган график расм чизиш учун мўлжалланган форма ёки сиртни тўла эгаллаши учун керак бўлади. Масалан, агар бирор *f(x)* функция 0 дан 1000 гача бўлган қийматларни қабул қилса, ҳамда унинг графигини чиқариш учун баландлиги 250 пиксел бўлган форма соҳаси олинган бўлса, у ҳолда Y ўқи бўйича масштабни формула билан топилади. Шундай қилиб, *f(x) =* 70 нуқтага координатаси Y = 233 бўлган нуқта мос келади. *Y* нинг координаталари

формула билан топилган. Бу ерда *h* – график қуриладиган соҳанинг баландлиги. Эътибор берган бўлсангиз, ифоданинг аниқ қиймати 232,5 га тенг. Аммо, нуқтани Canvas сиртига чиқариш учун фойдаланиладиган pixels ҳусусиятининг индекси фақат бутун сон бўлиши мумкин. Шунинг учун 232,5 сони ўзига энг яқин бутун сон, яъни 233 орқали яхлитланди.

Қуйида келтирилаётган дастур матни (11.4-листинг) pixels ҳусусиятидан фойдаланган ҳолда функциясининг графигини чизади. График чизиш учун форманинг мумкин бўлган барча қисми банд қилинади. Агар фойдаланувчи дастурнинг иши давомида ойна ўлчамларини ўзгартирса, график янги масштаб бўйича бошқатдан чизилади.

**11.5-листинг. Функциянинг графиги**

unit grfunc\_;

interface

uses Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs;

type

 TForm1 = class(TForm)

 procedure FormPaint(Sender: TObject);

 procedure FormResize(Sender: TObject);

 private

 { Private declarations }

 public

 { Public declarations }

 end;

var

 Form1: TForm1;

implementation

{$R \*.DFM}

⁄⁄ графигини чизиш талаб қилинган функция

 Function f(x:real):real;

 begin

 f := 2\*Sin(x)\*exp(x ⁄ 5);

 end;

⁄⁄ функция графигини чизиш

procedure GrOfFunc;

var

 x1,x2:real; ⁄⁄ функция аргументининг ўзгариш чегараси

 y1,y2:real; ⁄⁄ функция қийматининг ўзгариш чегараси

 x:real; ⁄⁄ функциянинг аргументи

 y:real; ⁄⁄ функцииянинг x нуқтадаги қиймати

 dx:real; ⁄⁄ аргумент орттирмаси

 l,b:integer; ⁄⁄ график соҳанинг чап қуйи бурчаги

 w,h:integer; ⁄⁄ график соҳанинг кенглиги ва баландлиги

 mx,my:real; ⁄⁄ X ва Y ўқлари бўйича масштаб

 x0,y0:integer; ⁄⁄ координаталар боши

begin

 ⁄⁄ график соҳа

 l := 10; ⁄⁄ X - чап юқори бурчакнинг координатаси

 b := Form1.ClientHeight-20;⁄ Y-чап юқори бурчак координатаси

 h := Form1.ClientHeight-40; ⁄⁄ баландлиги

 w := Form1.Width-40; ⁄⁄ кенглиги

 x1 := 0; ⁄⁄ аргумент диапазонининг қуйи чегараси

 x2 := 25; ⁄⁄ аргумент диапазонининг юқори чегараси

 dx := 0.01; ⁄⁄ аргумент қадами

 ⁄⁄ функциянинг [x1, x2]кесмадаги энг катта ва энг кичик қийматларини топамиз

 y1 := f(x1); ⁄⁄ минимум

 y2 := f(x1); ⁄⁄ максимум

 x := x1;

 repeat

 y := f(x);

 if y < y1 then y1 := y;

 if y > y2 then y2 := y;

 x := x+dx;

 until (x>=x2);

⁄⁄ масштабни ҳисоблаймиз

 my := h ⁄ abs(y2-y1); ⁄⁄ Y ўқи бўйича масштаб

 mx := w ⁄ abs(x2-x1); ⁄⁄ X ўқи бўйича масштаб

⁄⁄ ўқлар

 x0 := l;

 y0 := b-Abs(Round(y1\*my));

 with form1.Canvas do

 begin

 ⁄⁄ ўқлар

 MoveTo(l,b);LineTo(l,b-h);

 MoveTo(x0,y0);LineTo(x0+w, y0);

 TextOut(l+5, b-h, FloatToStrF(y2, ffGeneral, 6, 3));

 TextOut(l+5, b, FloatToStrF(y1, ffGeneral, 6, 3));

 ⁄⁄ графикни ясаш

 x := x1;

 repeat

 y := f(x);

 Pixels[x0+Round(x\*mx),y0-Round(y\*my)] := clRed;

 X := x + dx;

 until (x>=x2);

 end;

end;

procedure TForm1.FormPaint(Sender: TObject);

begin

 GrOfFunc;

end;

⁄⁄ дастур ойнасининг ўлчами ўзгарди

procedure TForm1.FormResize(Sender: TObject);

begin

 ⁄⁄ формани тозалаш

 form1.Canvas.FillRect(Rect(0,0,ClientWidth,ClientHeight));

 ⁄⁄ графикни ясаш

 GrOfFunc;

end;

end.

**Дастурни ишга тушириш**

Ушбу дастурда асосий ишни GrOfFunc процедураси бажаради. У дастлаб функциянинг [x1,x2] оралиқдаги максимал (у2) ва минимал (yl) қийматларини ҳисоблайди. Сўнгра форманинг кенглиги (Forml.Clientwidth -40) ва баландлиги (Form1.ClientHeight - 40) каби маълумотларни эътиборга олиб, X ўқи бўйича масштаб (mх) ва Y ўқи бўйича масштабларни (mу) топади.

График чиқариладиган сиртнинг баландлиги ва кенглиги форманинг ишчи (клиент) соҳаси сарлавҳа ва чегараларни ҳисобга олмасдан аниқланади. Масштаб топилганидан сўнг, процедура горизонтал ўқнинг ***у*** координатасини (OY) аниқлайди ва граифкнинг координата ўқларини чизади. Сўнгра бевосита функциянинг графигини ясашга ўтилади. (11.10-расм).

GrOfFunc процедурасига мурожаат қилишни onPaint ва onFormResize ходисаларни қайта ишлаш процедуралари бажаради. TForm1.FormPaint процедураси дастур ишга туширилиб, экранда форма пайдо бўлган заҳоти графикни чизилишини таъминлайди. TForm1.FormResize процедураси форма ўлчамлари ўзгарганидан кейин график ясашни ўз зиммасига олган.

11.10-расм. GrOfFunc процедураси билан чизилган график

Келтирилган ушбу дастур универсал ҳисобланади. Дастур матнидаги ***f(х)*** функцияни алмаштириб, бошқа функциянинг графигини ҳам ясаш мумкиш. Функциянинг кўринишидан қатъий назар, унинг графиги формани тўла эгаллайди.

**Эслатма:** Сизларга таклиф қилинган дастур графиги чизилаётган функция ҳам манфий, ҳам мусбат қийматлар қабул қилганда тўғри ишлайди. Агар сизнинг функциянгиз фақат мусбат, ёки фақат манфий қийматларни қабул қилса, дастур матнига кичик ўзгартиришлар киритишингизга тўғри келади. Бу ўзгаришларни аниқлаш ўзингизга ҳавола.

менюга

**11.5. Суратларни экранга чиқариш**

Сурат деганда биз олдиндан тайёрланган, (масалан, PAINT ёки бошқа матн муҳаррири ёрдамида) алоҳида файл сифатида ЭҲМ хотира қурилмаларидан бирида сақланаётган тасвирий файлларни назарда тутамиз.

Суратларни ***Image*** компонентаси ёрдамида расм чизиладиган сиртга чиқариш мумкин. Унинг нишони **Additional** қуроллар панелида жойлашган. (11.11-расм). Айниқса кенгайтмаси ***bmp, jpg*** ёки ***ico*** бўлган суратлар экранга осонгина чақирилади.

11.8-жадвалда Image компонентаси асосий ҳусусиятларининг рўйхати келтирилган.

 **11.11-расм.**  Image компонентасининг нишони

Image компонентасининг ҳусусиятлари 11.8-жадвал

|  |  |
| --- | --- |
| **Ҳусусияти**  | Мазмуни |
| Picture | Компонента майдонида акс эттириладиган расм  |
| Width, Height |  Компонентанинг ўлчамлари. Агар бу ўлчам сурат ўлчамдан кам бўлиб, AutoSize ва Strech лар False бўлса, у ҳолда суратнинг маълум бир қисмигина кўрсатилади.  |
| AutoSize | Компонентанинг ўлчамларини суратнинг ҳақиқий ўлчамларига боғлиқ равишда автоматик тарзда ўзгартирилиши |
| Strech | Суратни компонентанинг ўлчамларига боғлиқ равишда автоматик масштаблаштириш. Бунинг учун AutoSize нинг қиймати False бўлиши керак. |
| Vizible | Компонента ва сурат экрандаги форма сиртида кўринадиган бўлсинми? |

Компонета майдонига чиқариладиган суратларни илова формасини яратиш жараёнида ҳам, дастур ишлаётган вақтда ҳам кўрсатиш мумкин.

Илова формасини тайёрлаётган вақтда суратлар ***picture*** ҳусусиятига стандарт диалог ойнасидан фойдаланган ҳолда файлни танлаш орқали кўрсатилади. ***Picture***  ҳусусияти **Picture Editor** (11.12-расм) ойнасининг **Load** тугмаси чертилганда экранда пайдо бўлади. ***Image Editor*** ни ишга тушириш учун **Object Inspector** ойнасида ***Picture*** ва унинг ёнидаги устундан учта нуқтали тугма чертилади.

Агар суратнинг ўлчамлари компонента ўлчамларидан катта бўлса, у ҳолда strech ҳусусиятига True қийматини бериш ҳамда width ва Height ҳусусиятларига суратнинг ҳақиқий ўлчамларига пропорционал қийматларни бериш лозим.

Суратларни Image майдонига дастурнинг иши жараёни да чиқариш учун Picture ҳусусиятига LoadFromFile методини қўллаш лозим. Бу методнинг параметри сифатида сурат файлининг номи кўрсатилади. Масалан:

Form1.Image1.Picture.LoadFromFile('e:ғtempғbart.bmp')

кўрсатмаси суратни bart.bmp файлидан олади ва уни Image1 майдонига чиқаради.

LoadFromFile методи турли тасвирий форматдаги суратларни экранга чиқариш учун мўлжалланган: BMP, WMF, JPEG (jpg кенгайтмали файллар).

Қуйидаги дастур (унинг матни 11.5-листингда келтирилган) Image компонентасидан фойдаланувчи кўрсатган каталогдаги суратларни кўриш учун хизмат қилади. Бу дастурнинг диалог ойнаси 11.13-расмда кўрсатилган.

|  |  |
| --- | --- |
|  |  |
| 11.12-расм. Picture Editor ойнаси | **11.13-расм.** Слайд-проектор |

**11.5-листинг. Слайд-проектор**

unit shpic\_;

interface

uses Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs, ExtCtrls, {jpeg,} StdCtrls, Menus;

type

 TForm1 = class(TForm)

 Image1: TImage;

 Button1: TButton;

 Label1: TLabel;

 Edit1: TEdit;

 procedure FormActivate(Sender: TObject);

 procedure Button1Click(Sender: TObject);

 private

 { Private declarations }

 public

 { Public declarations }

 end;

var

 Form1: TForm1;

 aSearchRec : TSearchRec;

 aPath : String; ⁄⁄ сурат жойлашган каталог

 aFile : String; ⁄⁄ сурат файли

 n: integer = 0;

 iw,ih: integer; ⁄⁄ Image компонетасининг дастлабки ўлчами

implementation

{$R \*.DFM}

⁄⁄ сурат чиқариладиган соҳани сурат ўлчамига мослаб ўзгартириш

Procedure ScaleImage;

var

 pw, ph : integer; ⁄⁄ сурат ўлчами

 scaleX, scaleY : real; ⁄⁄ X ва Y бўйича масштаб

 scale : real; ⁄⁄ масштаб

begin

 ⁄⁄ сурат юкланди. Унинг ўлчамларини аниқлаймиз

 pw := Form1.Image1.Picture.Width;

 ph := Form1.Image1.Picture.Height;

 if pw > iw ⁄⁄ сурат кенглиги компонента кенглигидан катта

 then scaleX := iw ⁄ pw ⁄⁄ масштаблаштириш лозим

 else scaleX := 1;

 if ph > ih ⁄⁄ сурат баландлиги компонента баландлигидан катта

 then scaleY := ih ⁄ ph ⁄⁄ масштаблаштириш лозим

 else scaleY := 1;

 ⁄⁄ энг кичик коэффициент танлаймиз

 if scaleX < scaleY

 then scale := scaleX

 else scale := scaleY;

 ⁄⁄ сурат чиқариладиган соҳа ўлчамини ўзгартирамиз

 Form1.Image1.Height:=Round(Form1.Image1.Picture.Height\*scale);

 Form1.Image1.Width:=Round(Form1.Image1.Picture.Width\*scale);

 ⁄⁄ Strech = True ҳамда соҳа ўлчами сурат ўлчамига пропорционал

 ⁄⁄ бўлгани учун, сурат бузилишларсиз масштаблаштирилади

end;

⁄⁄ биринчи суратни чиқариш

procedure FirstPicture;

var

 r : integer; ⁄⁄ файлни қидириш натижаси

begin

 aPath := Form1.Edit1.Text;

 r := FindFirst(aPath + '\*.bmp', faAnyFile, aSearchRec);

 if r = 0 then

 begin

 aFile := aPath + aSearchRec.Name;

 Form1.Image1.Picture.LoadFromFile(aFile); ⁄⁄ суратни юклаш

 ScaleImage;

 r := FindNext(aSearchRec); ⁄⁄ навбатдаги файлни топиш

 if r = 0 then ⁄⁄ яна суратлар борми?

 Form1.Button1.Enabled := True;

 end;

end;

⁄⁄ навбатдаги суратни чиқариш

Procedure NextPicture();

var

 r : integer;

begin

 aFile := aPath + aSearchRec.Name;

 Form1.Image1.Picture.LoadFromFile(aFile);

 ScaleImage;

 ⁄⁄ навбатдаги суратни чиқаришга тайёрлаймиз

 r := FindNext(aSearchRec); ⁄⁄ навбатдаги файли қидирамиз

 if r <> 0

 then ⁄⁄ бошқа суратлар йўқ

 Form1.Button1.Enabled := False;

end;

procedure TForm1.FormActivate(Sender: TObject);

begin

 Image1.AutoSize := False;

 Image1.Stretch := True; ⁄⁄ масштаблашга рухсат берамиз

⁄⁄ суратларнинг дастлаб чиқариш соҳаси ўлчамларини эслаб қоламиз

 iw := Image1.Width;

 ih := image1.Height;

 Button1.Enabled := False; ⁄⁄ Навбадаги тугмасига рухсат йўқ

 FirstPicture; ⁄⁄ биринчи суратни чиқарилсин

end;

⁄⁄ Навбатдаги тугмаси чертилганда

procedure TForm1.Button1Click(Sender: TObject);

begin

 NextPicture;

end;

end.

**Дастурни ишга тушириш**

Дастур чиқарилаётган суратларни бузилишларсиз масштаб лаштиради. Буни тўғридан-тўғри Strech ҳусусиятига True қийматини бериб, эришиб бўлмайди. Биринчи ва қолган суратларни юклаш ва чиқаришни мос равишда FirstPicture ва NextPicture процедуралари бажаради. FirstPicture процедураси FindFirst функциясини биринчи учраган BMP-файлнинг номини топиш учун фойдаланади. FindFirst функциясининг параметри сифатида суратлар жойлашган каталог номи, asearchRec структураси (унинг Name майдони қидирилган маълумот топилган бўлса, талабни қаноатлантирувчи файл номини сақлайди) ҳамда сурат файли ниқобини кўрсатилади. Агар FindFirst функцияси чақирилганда кўрсатилган ҳеч бўлмаганда битта BMP-файл мавжуд бўлса, функциянинг қиймати нолга тенг. Бундай ҳолда LoadFromFile методи сурат файлини юклайди, ўнгра ScaleImage функцияси ишга туширилади. У компонента ўлчамини сурат ўлчамига мослаштиради. Юкланган суратнинг ўлчамини Form1.Image1.Picture.Width ва Form1.Image1.Picture.Height ҳусусияти қийматларига мурожаат қилиб, билиш мумкин. Уларнинг қийматлари Image компонентасининг ўлчамларига боғлиқ эмас.

менюга

**11.6. Битли тасвирлар**

Графика билан ишлаганда TBitMap (битли тасвир) типидаги объектлар иблан ишлаш жуда ҳам қулай. Битли тасвир компьютер хотирасида сақланаётган, демак кўринмайдиган сирт ҳисобланади. Унинг устида дастур турли тасвирларни ҳосил қилади. Битли тасвирларни (расмчаларни) осонлик ва тезлик билан форма ёки суратларни чиқариш соҳасига (image) чиқариш мумкин. Шунинг учун дастурда битли тасвирлардан унчалик катта бўлмаган расмлар, масалан, буйруқли тугмалар устидаги расмларни сақлаш учун қулай ҳисобланади.

Керакли тасвирларни битли тасвирга юклаш LoadFromFlie методи билан амалга оширилади, параметр сифатида керакли сурат сақланаётган BMP-файлининг номи кўрсатилади. Масалан, агар дастурда TBitMap типидаги pic ўзгарувчиси эълон қилинган бўлса,

***pic.LoadFromFiie('е : ⁄ images ⁄ aplane.bmp')***

буйруғи бажарилганидан сўнг, pic битли тасвири самолёт тасвирига эга бўлади.

Битли ттасвирларни форма ёки суратларни чиқариш соҳаси сиртига чақариш учун (canvas) сиртининг мос ҳусусиятига қўлланадиган Draw методидан фойдаланилади. Масалан.

Image1.Canvas.Draw(x,у, bm)

буйруғи Image1 компонентасининг сиртига bm битли тасвирини чиқаради ( бунда х ва у параметрлар расмнинг чап юқори бурчагининг коммпонента сиртидаги ҳолатини белгилайди).

Агар Draw методини қўллашдан аввал TBitMap объектининг Transparent ҳусусиятига True қиймати берилса, у ҳолда расмнинг чап қуйи бурчагидаги ранг билан бир ҳил рангга эга бўлган расм бўлаклари экранда кўрсатилмайди, уларнинг ўрнида фон кўриниб туради. Агар "шаффоф" ранг сифатида расмнинг чап қуйи бурчагидаги рангдан бошқа ранг олинса, у ҳолда Transparentcolor ҳусусиятига керакли рангни кўрсатувчи белгили константани қиймат қилиб бериш лозим.

Қуйидаги (11.6-листингда берилган) дастур бир нечта қисмдан иборат битли тасвирларни ҳосил қилиш усулини намойиш қилади.

**11.6-листинг. Битли тасвирлардан фойдаланиш**

unit aplane0;

interface

uses Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms, Dialogs;

type

 TForm1 = class(TForm)

 procedure FormPaint(Sender: TObject);

 private

 { Private declarations }

 public

 { Public declarations }

 end;

var

 Form1: TForm1;

 sky,aplane: TBitMap; ⁄⁄ битли тасвирлар: осмон ва самолёт

implementation

{$R \*.dfm}

procedure TForm1.FormPaint(Sender: TObject);

begin

⁄⁄ битли тасвирни яратиш

sky := TBitMap.Create;

aplane := TBitMap.Create;

⁄⁄ расмларни юклаш

sky.LoadFromFile('sky.bmp');

aplane.LoadFromFile('aplane.bmp') ;

Form1.Canvas.Draw(0,0,sky); ⁄⁄ фон

Form1.Canvas.Draw(20,20,aplane); ⁄⁄ чап самолётни чизиш

aplane.Transparent := True;

⁄⁄ Энди ранги битли тасвирнинг чап қуйи бурчаги ранги билан

⁄⁄ бир хил бўлган расм парчалари кўринмайди

Form1.Canvas.Draw(120,20,aplane); ⁄⁄ ўнг самолётни чизиш

⁄⁄ хотирани тозалаш

sky.free; aplane.free;

end;

end.

Дастур ишга тушганидан сўнг, илова ойнасида (11.14-расм) осмонда учаётган икки самолёт тасвири пайдо бўлади. Фон ва самолёт расмлари – битли тасвирлар ва улар файллардан юкланади. Чап самолёт атрофидаги оппоқ майдон aplane битли тасвирининг ҳақиқий ўлчамларини кўрсатади. Ўнг самолёт атрофида эса бундай майдон йўқ. Чунки, уни чиқаришдан аввал битли тасвирнинг Transparent ҳусусиятига True қиймати берилган.

 **11.14-расм. Transparent** ҳусусиятини расмга таъсири

**11.7. Мультипликация**

Мультипликация деганда биз одатда харакатланиб ва ўзгариб турадиган расмларни тушунамиз. Энг содда ҳолатда расм харакат қилиши ёки ўзгариши мумкин.

Юқорида айтиб ўтилдики, расмлар содда график элементлардан иборат бўлади. Расмни сирт бўйлаб сурилишини таъминлаш жуда ҳам осон: дастлаб расмни экранга чиқариш, маълум бир муддат ўтганидан сўнг уни ўчириш, расмни навбатдаги позициядан яна экранга чиқариш ва х.к. Расмни экранга чиқариш ва ўчириш орасидаги вақтни ҳамда расмнинг "эски" ва "янги" позициясини шундай танлаш мумкинки, кўраётган одамда расм экран бўйлаб харакатланаётгандай тасаввур қолдиради. Қуйидаги дастур (11.7-листинг) ва форманинг кўриниши (11.15-расм) илова ойнасининг чап томонидан ўнг томонига харакат қилишини намойиш қилади.

**11.15-расм.**  Харкатланаётган айлана

**11.8-листинг. Ҳаракатланаётган айлана**

unit Unit1;

interface

uses Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms, Dialogs, ExtCtrls;

type

 TForm1 = class(TForm)

 Timer1: TTimer;

 procedure Timer1Timer(Sender: TObject);

 procedure FormActivate(Sender: TObject);

 private

 { Private declarations }

 public

 { Public declarations }

 end;

var

 Form1: TForm1;

implementation

{$R \*.dfm}

var

x,y: byte; ⁄⁄ айлана марказининг координаталари

dx: byte; ⁄⁄ х координатасининг сурилиши

⁄⁄ ўчириш ва янги жойда айлана чизиш

procedure Ris;

begin

⁄⁄ айланани ўчириш

form1.Canvas.Pen.Color:=form1.Color;

form1.Canvas.Ellipse(x, y, x + 30, y + 30);

x := x + dx;

⁄⁄ айланани янги жойдан чизиш

form1.Canvas.Pen.Color := clBlack;

form1.Canvas.Ellipse(x, y, x + 30, y + 30) ;

end;

procedure TForm1.Timer1Timer(Sender: TObject);

begin

 Ris;

end;

procedure TForm1.FormActivate(Sender: TObject);

begin

x := 0; y := 10; dx := 5;

timer1.Interval := 50;

⁄⁄ OnTimer ходисасининг рўй бериши -0.5 сек

form1.canvas.brush.color := form1.color;

end;

end.

**Дастурни ишга тушириш**

Ушбу дастурда асосий вазифани Ris процедураси бажаради. У айланани ўчиради ва янги позициядан айлана чизади. Айланани ўчириш масаласи шу айланани фон ранги билан бир ҳил рангда устидан чизиш ҳисобига ҳал қилинади.

Илова формасига Ris процедурасини даврий равишда чақириш учун формага кўринмайдиган Timer (таймер) компонентаси киритилган. Унинг нишони **System** қуроллар панелида жойлашган. (11.16-расм). Timer компонетасининг ҳусусиятлари 11.9-жадвалда келтирилган.

**11. 16-расм.**  Timer компонентасининг нишони

Timer компонентасини формада одатдаги усуллар билан жойлаштириш мумкин, аммо, бу компонента новизуал, яъни кўринмайдиган бўлгани учун, дастурнинг иши давомида экранга чиқарилмайди. Шунинг учун уни форманинг ихтиёрий жойига ўрнатиш мумкин.

Timer компонентасининг ҳусусиятлари 11.9-жадвал

|  |  |
| --- | --- |
| Ҳусусияти  | Мазмуни |
| Name  | Компонентанинг номи. Компонентага мурожаат қилиш учун фойдаланилади. |
| Interval | OnTimer ходисасининг генерация қилиш даври. Миллисекундларда берилади |
| Enabled | Ишлашга рухсат. OnTimer ходисасини генерациясига рухсат бор (қиймати True) ёки йўқ (қиймати False) |

Timer компонентаси OnTimer ходисасини қайд қилади. Бу ходисанинг рўй бериши миллисекундларда ўлчанади ва Interval ҳусусиятининг қиймати билан аниқланади. Enabled ҳусусиятига эътибор беринг. У дастурга таймерни "ишга тушириш" ёки "тўхтатиб қўйиш" га рухсат беради. Агар Enabled ҳусусияти False га тенг бўлса, OnTimer ходисаси рўй бермайди .

OnTimer ҳодисаси юқоридаги дастурда Timer1Timer процедураси билан қайта ишланади. Бу процедура ўз навбатида Ris процедурасини ишга туширади. Бу дастурда Ris процедурасини даврий равишда, вақти-вақти билан ишга тушириш механизми қўлланган.

**Эслатма: *х*** ва ***у*** ўзгарувчилар (айлана марказининг координаталари) ва dx (айлананинг харакатланишида х координатани сурилиши) Ris процедурасидан ташқарида эълон қилинган, яни улар глобал ўзгарувчилар ҳисобланади. Шунинг учун уларнинг инициализация қилишни унутмаслик керак. (дастурда глобал ўзгарувчиларнинг инициализациясини FormActivate процедурасида амалга оширилади).

менюга

**11.8. Базавий нуқта методи**

Мураккаб, яъни бир нечта элементлардан иборат тасвирлар билан ишлаганда базавий нуқта методи деб аталадиган метод фойдаланилади. Бу методнинг ғояси қуйидагича:

1. Тасвирининг бирор нуқтасини базавий нуқта деб танлаб олинади.

**11.17-расм. Б**азавий нуқтага нисбатан тасвир координаталарини аниқлаш

2. Қолган нуқталарнинг координаталари базавий нуқтага нисбатан қайта ҳисобланади.

3. Агар тасвирдаги нуқталарнинг координаталарини бирликларга нисбатан (пикселларда эмас) қайта ҳисобланса, у ҳолда тасвирларни масштаблаштириш (мослаштириш) имконияти юзага келади.

11.7-расмда кемача тасвири берилган. Базавий деб (X0, Y0) координатали нуқтани оламиз. Қолган нуқталар координаталарини шу нуқтага нисбатан қайта ҳисоблаймиз 11.8-листингда "сузаётган" кемача тасвири ҳосил қилинган дастур матни берилган.

**11.8-листинг. Кемача**

unit ship\_;

interface

uses Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs, StdCtrls, ExtCtrls;

type

 TForm1 = class(TForm)

 Timer1: TTimer;

 procedure Timer1Timer(Sender: TObject);

 procedure FormActivate(Sender: TObject);

 private

 { Private declarations }

 public

 { Public declarations }

 end;

var

 Form1: TForm1;

 x,y: integer; ⁄⁄ кема координаталари (базавий нуқта)

implementation

{$R \*.DFM}

procedure Titanik(x,y: integer; ⁄⁄ базавий нуқта координаталари

 color: TColor); ⁄⁄ кема ранги

const

 dx = 5; dy = 5;

var

 buf: TColor;

begin

 with form1.canvas do

 begin

 buf := pen.Color; ⁄⁄ жорий рангни сақлаймиз

 pen.Color := color; ⁄⁄ керакли рангли ўрнатамиз

 ⁄⁄ чизамиз ... ⁄⁄ кема корпусини

 MoveTo(x, y);

 LineTo(x, y-2\*dy);

 LineTo(x + 10\*dx, y-2\*dy);

 LineTo(x + 11\*dx, y-3\*dy);

 LineTo(x + 17\*dx, y-3\*dy);

 LineTo(x + 14\*dx, y);

 LineTo(x, y);

 ⁄⁄ кема устидагилар

 MoveTo(x + 3\*dx, y-2\*dy);

 LineTo(x + 4\*dx, y-3\*dy);

 LineTo(x + 4\*dx, y-4\*dy);

 LineTo(x + 13\*dx, y-4\*dy);

 LineTo(x + 13\*dx, y-3\*dy);

 MoveTo(x + 5\*dx, y-3\*dy);

 LineTo(x + 9\*dx, y-3\*dy);

 ⁄⁄ капитан кўприкчаси

 Rectangle(x + 8\*dx, y-4\*dy, x + 11\*dx, y-5\*dy);

 ⁄⁄ труба

 Rectangle(x + 7\*dx, y-4\*dy, x + 8\*dx, y-7\*dy);

 ⁄⁄ иллюминаторлар

 Ellipse(x + 11\*dx, y-2\*dy, x + 12\*dx, y-1\*dy);

 Ellipse(x + 13\*dx, y-2\*dy, x + 14\*dx, y-1\*dy);

 ⁄⁄ мачта

 MoveTo(x + 10\*dx, y-5\*dy);

 LineTo(x + 10\*dx,y-10\*dy);

 ⁄⁄ оснастка

 MoveTo(x + 17\*dx, y-3\*dy);

 LineTo(x + 10\*dx, y-10\*dy);

 LineTo(x, y-2\*dy);

 pen.Color := buf; ⁄⁄ қаламнинг эски рангини тиклаймиз

 end;

end;

⁄⁄ таймер сигналини қайта ишлаш

procedure TForm1.Timer1Timer(Sender: TObject);

begin

 Titanik(x, y, form1.color); ⁄⁄ расмни ўчириш

 if x < Form1.ClientWidth

 then x := x + 5

 else begin ⁄⁄ янги рейс

 x := 0;

 y := Random(50) + 100;

 end;

 Titanik(x,y,clWhite); ⁄⁄ янги нуқтадан чизиш

end;

procedure TForm1.FormActivate(Sender: TObject);

begin

 x := 0; y := 100;

 Form1.Color := clNavy;

 Timer1.Interval := 50; ⁄⁄ таймер сигнали хар 50 мСек да

end;

end.

**Дастурни ишга тушириш**

Кемача тасвирини чизиш ва ўчириш вазифаларин Titanik процедураси бажаради. У параметрлар сифатида базавий нуқта координаталари ва кемача тасвирини чизиш учун рангни олади. Агар процедура чақирилганда ранг форма фони рангидан фарқ қилса, у ҳолда кемача чизилади, акс ҳолда ўчирилади. Titanik процедурасида dx ва dy константалар эълон қилинган бўлиб, улар тасвир нуқталари координаталарини ҳисоблашдаги қадамни (пикселларда) билдиради. Бу константаларнинг қийматларини ўзгартириб, тасвирни масштаблаштиришни амалга ошириш мумкин.

менюга

**11.9. Битли тасвирлардан фойдаланиш**

Аввалги пунктда бир нечта содда график элементлардан ташкил топган тасвирни кўрган эдик. Энди, битта мураккаб тасвирни бошқа мураккаб тасвир фонида харакатланишини кўрайлик. Масалан: шаҳар устидан ўтаётган самолёт тасвирини ясашга уриниб кўрамиз.

Расмнинг харакатланиш эффектини аввалги позициясига нисбатан расмни даврий равишда расмнинг қайта чизилиши (бир оз сурилиши) тарзида ифодалаймиз. Бунда, расмни ҳам гал қайта чизишдан аввал, унинг олдинги ҳолатини ўчириш лозим бўлади. Расмни ўчириш эса фон-расм ёки унинг харакатланаётган расм билан тўсилган жойини қайта чизиш орқали амалга оширилади.

Кўрилаётган дастурда иккинчи усулни қўллаймиз. Расмлар ёрдамида Imagt компонентасиниг Canvas ҳусусиятига Draw методини қўллаб, форма сиртига чиқарилади. Ўчириш эса нусха олиш усули (copyRect методи) билан буфердан керакли фонни Image компонетаси сиртига чиқариш билан ҳал қилинади. Бу дастурнинг формаси 11.18-расмда, матни эса 11.9-листингда келтирилган.

Image компонентаси фонни ҳосил қилиш учун фойдаланилади, Timer компонентаси эса самолёт тасвирини ўчириш ва янги жойдан экранга чиқариш жараёнини даврий равишда бўлишини таъминлайди

**11.9-листинг. Учаётган самолёт**

unit aplane\_;

interface

uses Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs, ExtCtrls, StdCtrls, Buttons;

type

 TForm1 = class(TForm)

 Timer1: TTimer;

 Image1: TImage;

 procedure FormActivate(Sender: TObject);

 procedure Timer1Timer(Sender: TObject);

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

 private

 { Private declarations }

 public

 { Public declarations }

 end;

var

 Form1: TForm1;

implementation

{$R \*.DFM}

var

 Back, bitmap, Buf : TBitMap; ⁄⁄ фон, картинка, буфер

 BackRct : TRect; ⁄⁄ буфердан тикланадиган фон соҳаси

 BufRct: Trect; ⁄⁄ Фонни тиклаш учун керак бўлган буфер соҳаси

 x,y:integer; ⁄⁄ расмнинг жорий ҳолати

 W,H: integer; ⁄⁄ расмнинг ўлчамлари

procedure TForm1.FormActivate(Sender: TObject);

begin

 ⁄⁄ учта объект -битли тасвир яратиш -

 Back := TBitmap.Create; ⁄⁄ фон

 bitmap := TBitmap.Create; ⁄⁄ расм

 Buf := TBitmap.Create; ⁄⁄ буфер

 ⁄⁄ фонни юклаш ва чиқариш

 Back.LoadFromFile('factory.bmp');

 Form1.Image1.canvas.Draw(0,0,Back);

 ⁄⁄ харакатланадиган расмни юклаш

 bitmap.LoadFromFile('aplane.bmp');

 ⁄⁄ "шаффоф" рангни танлаш

 bitmap.Transparent := True;

 bitmap.TransParentColor := bitmap.canvas.pixels[1,1];

 ⁄⁄ расм қўйиладиган фон соҳаси нусхасини сақлаш учун буфер яратиш

 W := bitmap.Width;

 H := bitmap.Height;

 Buf.Width := W;

 Buf.Height := H;

 Buf.Palette := Back.Palette; ⁄⁄ палитралар мослигини таъминлаш учун !!

 Buf.Canvas.CopyMode := cmSrcCopy;

 ⁄⁄ фонни тиклаш учун зарур бўлган буфер соҳасини аниқлаймиз

 BufRct := Bounds(0, 0, W, H);

 ⁄⁄ расмнинг бошланғич ҳолати

 x := -W;

 y := 20;

 ⁄⁄ фоннинг сақланадиган қисмини аниқлаймиз

 BackRct := Bounds(x, y, W, H);

 ⁄⁄ ва уни сақлаймиз

 Buf.Canvas.CopyRect(BufRct, Back.Canvas, BackRct);

end;

⁄⁄ таймер сигналини қайта ишлаш

procedure TForm1.Timer1Timer(Sender: TObject);

begin

 ⁄⁄ буфердан фонни тиклаб, расмни ўчирамиз

 Form1.image1.canvas.Draw(x,y,Buf);

 x := x + 2;

 if x>form1.Image1.Width then x := -W;

 ⁄⁄ сақланадиган фон соҳасини аниқлаймиз

 BackRct := Bounds(x, y, W, H);

 ⁄⁄ унинг нусхасини сақлаймиз

 Buf.Canvas.CopyRect(BufRct, Back.Canvas, BackRct);

 ⁄⁄ расмни чиқарамиз

 Form1.image1.canvas.Draw(x,y,bitmap);

end;

⁄⁄ дастур ишини якунлаймиз

procedure TForm1.FormClose(Sender: TObject; var Action: TCloseAction);

begin

 ⁄⁄ битли тасвир учун ажратиллган хотирани бўшатамиз

 Back.Free;

 bitmap.Free;

 Buf.Free;

end;

end.

**Дастурни ишга тушириш**

|  |  |
| --- | --- |
|  |  |
| 11.19-расм. Самолёт дастурининг фони ва иш вақтидаги кўриниши |

Фон ва самолётнинг битли тасвирларини, шунингдек самолёт тўсиб қолаётган фон соҳасини сақлаш учун TbitMap типидаги объектлардан фойдаланнлади. Бу объектлар FormActivate процедураси ёрдамида динамик равишда ҳосил қилинади. Шу процедуранинг ўзи фон расми бўлган factory.bmp ҳамда самолётнинг расми aplane.bmp файлларини юклайди, ҳамда самолёт биринчи марта қўйиладиган фон соҳасини сақлаб қолади.

Фоннинг нусхасини сақлаш CopyRect методи ёрдамида амалга оширилади. У битта тўғри тўртбурчак битли тасвир нусхасини бошқасига кўчиришга имкон беради. CopyRect методи қўлланаётган объект битли тасвирни қабул қилувчи бўлади. Методнинг параметрлари нусха кўчириладиган соҳанинг координаталари ва ўлчами, нухса олинадиган сирт, нусҳа олинаётган соҳанинг ҳолати ва ўлчамларидан иборат бўлади. Буферга кўчирилаётган самолёт расми қўйиладиган фон соҳасининг ҳолати ва ўлчами ҳамда буфердан қайта тикланадиган фон соҳаси ҳақидаги маълумотлар Trect типидаги BackRct структурасида сақланади. Бу структурани тўлдириш учун Bounds функциясидан фойдаланилади.

Эътибор беринг, харакатланадиган битли тасвирнинг чап юқори бурчагининг х – координатаси манфий сон ва у битли тасвир кенглигига тенг. Шунинг учун дастур ишининг бошланишида самолёт тасвири кўринмайди. Расмни чизиш кўринадиган соҳадан ташқарида бошланади. Ҳар бир OnTimer ходисаси рўй берганда ***x-***нинг координатаси ўсиб боради ва экранда битли тасвирининг координаталари нолдан катта бўлган қисми пайдо бўлади. Шундай қилиб, фойдаланувчида самолёт ҳудди ойнанинг чап чегарасидан учиб чиқаётгандай тасаввур ҳосил бўлади.

менюга

**11.10. Дастур ресурсидан битли тасвирларни юклаш**

11.9-листингда келтирилган фон ва самолёт расмлари файллардан юкланади. Бу ҳар доим ҳам қулай бўлавермайди. Delphi тили барча зарурий битли тасвирларни ресурслар кўринишида бажарилувчи дастурнинг файлига қўшиб қўйишга имкон беради. Бу тасвирлардан кейинчалик эхтиёжга қараб ресурсдан, яъни бажарилувчи (EXE-файл) файлдан юклаб мумкин бўлади.

**Ресурслар файлини яратиш.** Битли тасвирларни ресурсдан юклаш мумкин бўлиши учун аввал, барча битли тасвирларни ўз ичига олган ресурслар файлини яратиш лозим бўлади.

Ресурслар файлини **Image Editor** утилити (тасвирлар муҳаррири) ёрдамида ташкил қилиш мумкин. Бу утилит **Tools** менюсидаги **Image Editor** буйруғини танлаш орқали ишга туширилиши мумкин.

Янги ресурслар файлини яратиш учун **File** менюсидан **New** буйруғини танлаш керак, сўнгра экранда пайдо бўладиган остменюдан - **Resource File** (ресурслар файли) тугмасини чертилади. Натижада янги ресурслар файлининг ойнаси пайдо бўлади, **Image Editor** ойнасининг менюлар сатрида янги пункт - **Resource** пайдо бўлади.

Бу файлга янги ресурсни қўшиш учун **Resourse** менюсидан **New** буйруғини, сўнгра очиладиган рўйхатдан ресурс типи танланади. Бизнинг ҳолда **Bitmap** (битли тасвир) танланади. Бу танловдан кейин **Bitmap Properties** (битли тасвирнинг ҳусусияти) диалог ойнаси экранга узатилади. Ундан фойдаланиб яратиладиган расмларнинг ўлчами ва ранглар сони белгиланади. ОК тугмасини босиш билан **Bitmap Properties** диалог ойнасига тармоқланган рўйхатдан **Bitmap1** элементи чақирилади. Бу элемент файлга қўшилган янги ресурсга мос келади.

**Bitmap1** — бу автоматик тарзда яратилган ресурс номи бўлиб, **Resourse** менюсидаги **Rename** буйруғини танлаш ва сўнгра керакли номли ёзиш билан ўзгартирилиши мумкин. **Bitmap1** номини ўзгартирилганидан кейин битли тасвирни яратишга киришиш мумкин. Бунинг учун **Resourse** менюсидан **Edit** буйруғини танлаш лозим. Натижада график муҳаррирнинг таҳрирлаш ойнаси очилади.

**Image Editor** график редактори дастурчига шундай редакторларга ҳос бўлган қуроллар мажмуасини таклиф қилади. Бу қуроллардан фойдаланиб дастурчи ўзи учун зарур бўлган тасвирларни ясаши мумкин. Агар иш вақтида расм масштабини ўзгартиришга эхтиёж пайдо бўлса, масштабни катталаштириш учун **View** менюсидан **Zoom In** меню буйуруғи**, кичиклаштириш учун эса Zoom Out** буйруғи танланади. Тасвирни ҳақиқий масштабда кўриш учун эса **View** менюсидан **Actual Size** буйруғини танлаш лозим.

Агар керакли расм олдиндан алоҳида файл шаклида мавжуд бўлса, уни алмашув буфери (clipboard) ёрдамида ресурслар файлининг битли тасвирига жойлаш мумкин. Бу қуйидагича бажарилади:

1. Дастлаб график редактор, масалан Microsoft Paint ишга туширилади, унга тасвир файлини юкланади ва бу тасвирни тўла ёки маълум бир қисми ажратилади. Ажратиш жараёнида диққатни ажратилган соҳанинг пикселлардаги ўлчами ҳақидаги маълумотга қаратиш лозим. (Paint ажратилаётган соҳанинг ўлчамларини ҳолатлар сатрига чиқаради ). Сўнгра **Правка** менюсидан Копировать тугмаси танланади. Натижада тасвирнинг ажратилган қисми буфер хотирага тушади.

2. Сўнгра **Image Editor** га ўтиб, буфердаги тасвир қўшиладиган ресурс танланади ва ресурс характеристикаларини буфердаги тасвир характеристикаси қийматларига мосланади. Ресурс характеристикаси қийматлари **Bitmap Properties** диалог ойнасининг махсус ойна-майдонларига киритилади. Бу диалог ойнасини **BitMap** менюсининг **Image Properties** буйруғи билан экранга чақириш мумкин. Барча характеристикалар ўрнатилганидан сўнг **Edit** менюсидан Paste буйруғи билан буфер хотирадаги тасвирни ресурсга қўшиш мумкин.

3. Ресурслар файлига барча керакли ресурсларни қўшилганидан сўнг, ресурс файлини шу ресурслар учун мўлжалланган бажарилувчи дастур жойлашган каталогда сақлаб қўйиш лозим. Файл одатдаги усул билан сақлаб қўйилади. Image Editor муҳаррири бу файлга **res** кенгайтмасини беради.

**Ресурслар файлини ишга тушриш.** Ресурслардан дастурда фойдаланиш мумкин бўлиши учун, дастур матнида компиляторга бажарилувчи файлнинг таркибига ресурслар файлини ҳам қўшиб қўйиш ҳақида кўрсатма берилиши лозим. Бу кўрсатма умумий кўринишда қуйидагича ёзилади:

{**$R Ресурслар файли**}

Бу ерда **Ресурслар файли** — ресурслар файлининг номи. Масалан,

{$R images.res}

Ресурслар файлини бажариладиган файл таркибига қўшиш кўрсатмаси одатда модул матнининг бошланишида ёзилади.

**Эслатма:**  Агар модул файлининг ва ресурслар файлининг номлари бир ҳил бўлса, ресурс файлининг номи ўрнига "\*" белгисини қўшиш мумкин. Бу ҳолда юқоридаги кўрсатма қуйидагича ёзилади:

{$R \*.res}

Ресурсдан тасвирларни TBitMap типидаги ўзгарувчига юклаш масаласи LoadFromResourceName методи ёрдамида ҳал қилинади. Унинг иккита параметри бор: дастурнинг идентификатори ва ресурс номи. Дастур идентификатори сифатида Hinstance глобал ўзгарувчиси қўлланади. Ресурс номи сатрли константа тарзида ёзилади. Масалан, тасвирни ресурсдан Pic ўзгарувчисига юклаш учун

Pic.LoadFromResourceName(Hinstance,'FACTORY') ;

кўринишидаги буйруқдан фойдаланилади.

Намуна қилиб, матни 11.10-листингда келтирилган дастурни олиш мумкин. Бу дастурда фон ва самолёт татсвирлари ресурсдан юкланади.

**11.10-листинг. Тасвирларни ресурсдан юклаш учун намуна.**

unit aplane1\_;

{$R images.res} ⁄⁄ ресурслар файлини қўшиш

interface

uses Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs, ExtCtrls, StdCtrls, Buttons;

type

 TForm1 = class(TForm)

 Timer1: TTimer;

 Image1: TImage;

 procedure FormActivate(Sender: TObject);

 procedure Timer1Timer(Sender: TObject);

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

 private

 { Private declarations }

 public

 { Public declarations }

 end;

var

 Form1: TForm1;

 Back, bitmap, Buf : TBitMap; ⁄⁄ фон, расм, буфер

 BackRct, BufRct: TRect; ⁄⁄ фон, расм, буфер соҳалари

 x,y:integer; ⁄⁄ расмнинг чап юқори бурчаги координаталари

 W,H: integer; ⁄⁄ расмнинг ўлчами

implementation

{$R \*.DFM}

procedure TForm1.FormActivate(Sender: TObject);

begin

 Back := TBitmap.Create; ⁄⁄ фон

 bitmap := TBitmap.Create; ⁄⁄ расм

 Buf := TBitmap.Create; ⁄⁄ буфер

 ⁄⁄ ресурсдан фонни юклаш

 Back.LoadFromResourceName(HInstance,'FACTORY');

 Form1.Image1.canvas.Draw(0,0,Back);

 ⁄⁄ ресурсдан харакатланиши талаб қилинган расмни юклаш

 bitmap.LoadFromResourceName(HInstance,'APLANE');

 bitmap.Transparent := True;

 bitmap.TransParentColor := bitmap.canvas.pixels[1,1];

 ⁄⁄ устига расм тушадиган фон соҳасини сақлаш учун буфер яратиш

 W := bitmap.Width;

 H := bitmap.Height;

 Buf.Width := W;

 Buf.Height := H;

 Buf.Palette:=Back.Palette; ⁄⁄ палитралар мослигини таъминлаш учун

 Buf.Canvas.CopyMode := cmSrcCopy;

 BufRct := Bounds(0, 0, W, H);

 x := -W;

 y := 20;

 ⁄⁄ фоннинг сақланадиган соҳасини аниқлаш

 BackRct := Bounds(x,y,W,H);

 ⁄⁄ ва уни сақлааймиз

 Buf.Canvas.CopyRect(BufRct,Back.Canvas,BackRct);

end;

procedure TForm1.Timer1Timer(Sender: TObject);

begin

 ⁄⁄ фонни қайти таклиб, расмни ўчираиз

 Form1.image1.canvas.Draw(x,y,Buf);

 x := x + 2;

 if x>form1.Image1.Width then x := -W;

 ⁄⁄ Фоннинг сақланадиган соҳасини аниқлаймиз

 BackRct := Bounds(x, y, W, H);

 ⁄⁄ унинг нусхасини сақлаймиз

 Buf.Canvas.CopyRect(BufRct,Back.Canvas,BackRct);

 ⁄⁄ расмни чиқарамиз

 Form1.image1.canvas.Draw(x,y,bitmap);

end;

procedure TForm1.FormClose(Sender: TObject; var Action: TCloseAction);

begin

 Back.Free;

 bitmap.Free;

 Buf.Free;

end;

end.

Тасвирларни ресрурсдан юклашнинг афзаллиги катта: дастурларни бошқа компьютерларга ўтказишда ҳамма керакли файлларнинг мавжуд бўлиши ҳақида қайғурилмайди, чунки, дастур учун зарур бўлган барча файллар бажариладиган файл ичида сақланади.

менюга

**11.11. "Мультфильм" кўриш**

Ушбу пунктда қандай қилиб диалог ойнасида мультфильм намойиш қилиш мумкинлигини кўрамиз.

Телефон ва компьютер ўртасида югураётган қизил квадратча эффектини ҳосил қилиш масаласи диалог ойнасида алмашувчи расмларни навбатма-навбат чиқариш ҳисобига ҳал қилинади.

Мультфильм кадрлари одатда битта файлда ёки битта ресурсда жойлашган бўлади. Дастур ўз ишини бошлаганидан кейин улар буферга юкланади. Шунинг бу расмлар BitMap типидаги тасвирлар бўлса, мақсадга мувофиқ бўлади. Мультфильмни намойиш қилувчи (экранга чиқарувчи) процедуранинг вазифаси навбатдаги кадрни ажратиш ва уни форманинг керакли жойига қўйишдан иборат бўлади.

Кадрларни форма сиртига чиқариш шу форманинг Canvas ҳусусиятига copyRect методини қўллаш ҳисобига амалга оширилади. CopyRect методи берилган график сиртнинг тўғри тўртбурчак шаклидаги соҳасини бошқа сиртга кўчиради.

CopyRect методи умумий кўринишда қуйидагича ёзилади:

***Canvas1.CopyRect(Соҳа1, Canvas2, Соҳа2)***

Бу ерда ***canvas1*** — нусха кўчириладиган график сирт; ***Canvas2*** — нусхаси кўчириладиган график сирт; ***Соҳа2*** парметри – кўчириладигшан тўғри тўртбурчакл соҳанинг ҳолати ва ўлчамлари; ***Соҳа1*** параметри эса нусханинг Canvas1 сиртидаги ҳолати.

 Соҳа1 ва Соҳа2 параметрлари сифатида Trect типидаги структурадан фойдаланилади. Уларнинг майдонлари соҳанинг ҳолати ва ўлчамларини белгилайди. TRect структураси майдонларини Bounds функцияси ёрдамида тўлдирилиши мумкин. Бу функцияга мурожаат қилиш буйруғи қуйидагича:

Bounds(x,у,Width,Height)

Бу ерда ***х*** ва ***у*** —соҳанинг чап ва ўнг юқори координаталари; width ва Height — соҳанинг кенглиги ва баландлиги.

Матни 11.11-листингда берилган қуйидаги дастур диалог ойнасига Delphi устунини чиқаради ва унинг атрофида қандайдир объектнинг учиб айланаётганлигини тасвирлаайди. 11.19-расмда шу мультфильмни кадрлари келтирилган. (film.bmp файлидаги тасвир).

Дастурнинг диалог ойнаси 11.20-расмда кўрсатилган. У ягона омпонента-таймерни ўз ичига олган.

**11.18-расм.** Мультфильм кадрлари

|  |  |
| --- | --- |
|  |  |
| 11.20-расм. Дастурнинг формаси |

**11.11-листинг. Мультфилм (CopRect методидан фойдаланиш)**

unit multik\_;

interface

uses Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs, ExtCtrls, StdCtrls;

type

 TForm1 = class(TForm)

 Timer1: TTimer;

 procedure FormActivate(Sender: TObject);

 procedure Timer1Timer(Sender: TObject);

 private

 { Private declarations }

 public

 { Public declarations }

 end;

var

 Form1: TForm1;

implementation

{$R \*.DFM}

const

 FILMFILE = 'film2.bmp'; ⁄⁄ фильм - bmp-файл

 N\_KADR = 12; ⁄⁄ фильмдаги кадрлар сони (берилган файл учун)

var

 Film: TBitMap; ⁄⁄ фильм - ҳамма кадрлар

 Kadr: TBitMap; ⁄⁄ жорий кадр

 WKadr,HKadr: integer; ⁄⁄ кадрнинг кенглиги ва баландлиги

 CKadr: integer; ⁄⁄ жорий кадр номери

 RectKadr: TRect; ⁄⁄ кадрнинг фильмдаги ҳолати ва ўлчамларие

Rect1:Trect; ⁄⁄ фильмни кўрсатиш соҳасининг координата ва ўлчами

procedure TForm1.FormActivate(Sender: TObject);

begin

 Film := TBitMap.Create;

 Film.LoadFromFile(FILMFILE);

 WKadr := Round(Film.Width⁄N\_Kadr);

 HKadr := Film.Height;

 Rect1 := Bounds(10, 10, WKadr, HKadr);

 Ckadr := 0;

 Form1.Timer1.Interval := 150; ⁄⁄ кадрларни янгилаш даври

 Form1.Timer1.Enabled := True; ⁄⁄ таймерни ишга тушириш

end;

⁄⁄ кадрни кўрсатиш

procedure DrawKadr;

begin

 ⁄⁄ жорий кадрнинг филмдаги ўрнини топамиз

 RectKadr := Bounds(WKadr\*CKadr, 0, WKadr, HKadr);

 ⁄⁄ кадрни фильмдан чиқариш

 Form1.Canvas.CopyRect(Rect1,Film.Canvas,RectKadr);

 ⁄⁄ навбатдаги кадрни чиқаришга тайёргарлик

 CKadr := Ckadr + 1;

 if CKadr := N\_KADR

 then CKadr := 0;

end;

⁄⁄ таймер сигналини қайта ишлаш

procedure TForm1.Timer1Timer(Sender: TObject);

begin

 DrawKadr;

end;

end.

**Дастурни ишга тушириш**

Дастур учта процедурадан иборат. TForm1.FormActivate процедураси Film объектини ташкил қилади ва унга фильмнинг кадрлари ёзилган BMP-файлни юклайди, сўнгра, юкланган битли тасвир ҳақидаги маълумотлардан фойдаланган ҳолда кадрнинг баландлиги ва кенглигини белгилаб беради.

Шундан кейин жорий кадрни сақлаш учун мўлжалланган TBitMap типидаги Kadr объекти яратилади. Шуни ёдда тутиш керакки, Kadr объекти яратилганидан кейин width ва Height ҳусусиятларининг қийматлари аниқланади. Агар шундай қилинмаса, яратилган объект мавжуд, аммо битли тасвир учун хотирадан жой ажратилмайди. TForml. FormActivate ўз ишининг якунида жорий кадрнинг номерини белгилаб қўяди ва таймерни ишга туширади.

Дастурдаги асосий ишни, яъни фильмнинг навбатдаги кадрини ажратиш ҳамда уни формага чиқаришни DrawKadr процедураси бажаради. DrawKadr процедурасини бошқариш, ишга тушириш вазифасини OnTimer ҳодисасининг қайта ишловчи TForm1.Timer1Timer процедураси бажаради.

11.21-расм. Canvas1. CopyRect (Rect1, Canvas2, Rect2) буйруғи canvas сиртининг Rect1 соҳага Canvas2 сиртнинг Rect2 соҳасии кўчиради.

менюга