ÛЗБЕКИСТОН РЕСПУБЛИКАСИ
ВАЗИРЛАР МАҲКАМАСИ ҲУЗУРИДАГИ
ТОШКЕНТ ИСЛОМ УНИВЕРСИТЕТИ

ХОДЖИМУРАТОВА З. З.

“Delphida дастурлаш асослари”

“Delphida дастурлаш асослари” курсидан
ТИУ талабалари учун ўқув қўлланма

ТОШКЕНТ
“УНИВЕРСИТЕТ”
2007

Ушбу қўлланмада Delphi муҳити билан танишиш, унинг келиб чиқиши, версиялари, Delphi муҳитида дастурлаштириш тили бўлмиш Object Pascal тили, бу тилнинг структураси, алфавити, буйруқлари, маълумотлар тоифалари, объектли дастурлаштиришнинг 3 принципи, динамик кутубхоналар ҳосил қилиш ва уларни чақириб ишлатиш, Delphi муҳитининг асосий компонентлари, маълумотлар базалари ҳақида умумий тушунчалар, Delphi муҳитида маълумотлар базаси билан ишловчи компонентлер ва бошқа мавзулар ҳақида тушунчалар ёритилган. Барча мавзулар мисоллар билан баён этилган.

Ўқув қўлланма 5521900 – “Информатика ва ахборот технологиялари” йўналиши бўйича таҳсил олаётган талабаларга мўлжалланган бўлиб, ундан бошқа олий ўқув юрти талабалари, мустақил ўрганувчилар ҳам фойдаланишлари мумкин.

Тақризчилар:

Техника фанлари доктори, профессор Ш. Нуритдинов,
Техника фанлари номзоди Н. Турсунов

Тошкент ислом университети ўқув-услубий кенгаши мажлисининг “__”_____ 2007 йилдаги __-баённомаси билан нашрга тавсия қилинган.

ЗУХРА ЗАЙНИТДИНОВНА ХОДЖИМУРАТОВА

DELPHIДА ДАСТУРЛАШ АСОСЛАРИ

(ўқув қўлланма)

Муқаддима

Ҳозирги кунда Республикамиздаги олий ўқув юртларида «Информатика ва ахборот технологиялари» йўналиши талабаларига ўқув режасига кўра турли хил дастурлаш тилларини ўргатиш мўлжалланган. Улар орасида объектга йўналтирилган дастурлаштириш тиллари бўлмиш Visual Basic, Delphi, Visual C+ ва бошқаларни мисол қилиб келтириш мумкин.

Қўлланма Delphiда дастурлашни бошловчилар учун мўлжалланган бўлиб, ундан барча босқичдаги талабалар ва тилни мустақил ўрганаётганлар фойдаланишлари мумкин. Ушбу қўлланма Тошкент ислом университети базасида ишлаб чиқилди.

Мазкур фанни ўзлаштириш жараёнида талабалар бевосита компютер билан мулоқотда бўлиб, юқорида санаб ўтилган мавзулар бўйича амалий ва лаборатория машғулотларини бажаришлари керак.

DELPHIГА КИРИШ

Delphi мущити замонавий 32 разрядли графикали операцион тизимлар (ОТ) Windows 95, Windows 98, Windows NT, Windows 2000 (яъни windows32 деб юритилади) бош=арувида ишловчи дастурлар тузиш учун мылжалланган.

Олдинги MS-DOSда дастурлашга нисбатан Windows тизимида дастурлаш анча =ийинро=дир. Microsoft ва Borland корпорациялари Windows 3.1 пайдо былгандан бошлабо= мос воситалар ярата бошлади. Масалан, 1991 йил Turbo Pascal for Windows, 1992 йил бу дастурлаш тизимининг янгиланган версиясини Borland Pascal with Object 7.0 ни чи=арди. Бу дастлабки махсуслаштирилган инструментлар Windows нинг анчагина билимларини талаб =илар эди. 1993 йилга келиб Microsoft фирмаси биринчи визуал дастурлаш мущити - Visual Basicни ишлаб чи=ди щамда Windows учун дастурлаштириш бир мунча осонлашди. Бунга жавобан Borland 1995 йили Delphi нинг биринчи версиясини чи=арди, бир йилдан сынг иккинчи версияси, яна бир йилдан сынг учинчи версияси, сынгра тыртинчи ва бешинчи версияларини ишлаб чи=арди. Нищоят, 2001 йил май ойида олтинчи версиясини ишлаб чи=ди.

Барчага маълумки, MS-DOSда дастурлашни ырганиш ва эгаллаш учун Turbo Pascal тили хизмат =илади. Delphi эса Pascal ориентирлантирилган (йыналтирилган) дастурлаш воситалари сериясининг давомчиси былди щамда щозирги ва=тда Windows-дастурлаштириш учун энг =улай инструмент щисобланади.

Delphiнинг биринчи версияси 1995 йилнинг май ойида щали Windows 95 йы=лигида (лекин Windows NT мавжуд эди) пайдо былди. Бу 16 разрядли Windows 3.1 (3.11) бош=арувида ишловчи ягона версиядир. Унда биринчи марта объектларнинг янги модели ишлатиб кырилган былиб, турли объектли-йыналтирилган тиллар билан биргаликда (асосан C++ да) тузишга щаракат =иинган. Бу модель шу даражада \алаба =илдики, хатто Turbo Pascalда ыша пайтларда ишлатиладиган объектларнинг кераги былмай =олди (яъни у объектларни барчасини

янги модель ыз ичига =амраб олди). Янги модель эса «класслар» номини олди.

Класслар динамик хотирадан фаол фойдаланади, шунинг учун тилнинг баён этилиши бир мунча ызгарди щамда тилнинг номи Object Pascal деб номланди. Turbo Pascalга нисбатан Object Pascalга кыпгина =ышимчалар ва ызгартиришлар киритилди.

Биринчи версиянинг компонентлар кутубхонаси асосан маълумотлар базасини дастурлаштиришга йыналтирилганлигини бошидано= кырсатди. Шу ма=садда биринчи ва кейинги версиялари маълумотларга мурожаатни таъминловчи махсус инструмент BDE (Borland DataBase Engine - Borland копорациясининг маълумотлар базаси машинаси) билан щамда InterBase маълумотлар базаси сервери билан жищозланди. Биринчи версиядаги компонентлар палитраси 9 та сащифадан ва 79 компонентдан иборат.

Иккинчи ва кейинги версиялари 32 разрядли Windows ОТ бош=аруви асосида ишлаш учун мылжалланган. Унга кыпгина ызгартиришлар киритилган. Компонентлар палитрасининг 12 сащифасида 114 та стандарт компонентлар жойлаштирилди.

Учинчи версияси Delphiнинг ишончли версияларидан бири былиб, унга щам кыпгина ызгартиришлар киритилган. У тыртта комплектацияда чи=ди: Standart, Proffessional, Client/ServerSuite ва Enterprise. 13 та сащифада 148 та стандарт компонент жойлаштирилди.

Delphi 6.0 эса Delphi 5.0 дан кейин 2 йилдан сынг дунё юзини кырди. Чунки Delphi 6.0 ни ишлаб чи=ариш билан параллел равишда Delphiнинг Linux ОТ учун варианти устида щам ишлар олиб борилган. 2001 йил февралда Linux учун Delphi тузилди ва у Kylix деб номланди. Натижада Delphi 6.0 нинг 2 та ажойиб хусусияти пайдо былди: Windows 32 бош=арувида щам, Linux бош=арувида щам дастур яратиш имконияти ту\илди. Delphi 6.0 нинг яна бир хусусияти бу web-дастурлаш учун имкониятлар очилганлигидир (websnap архитектураси).

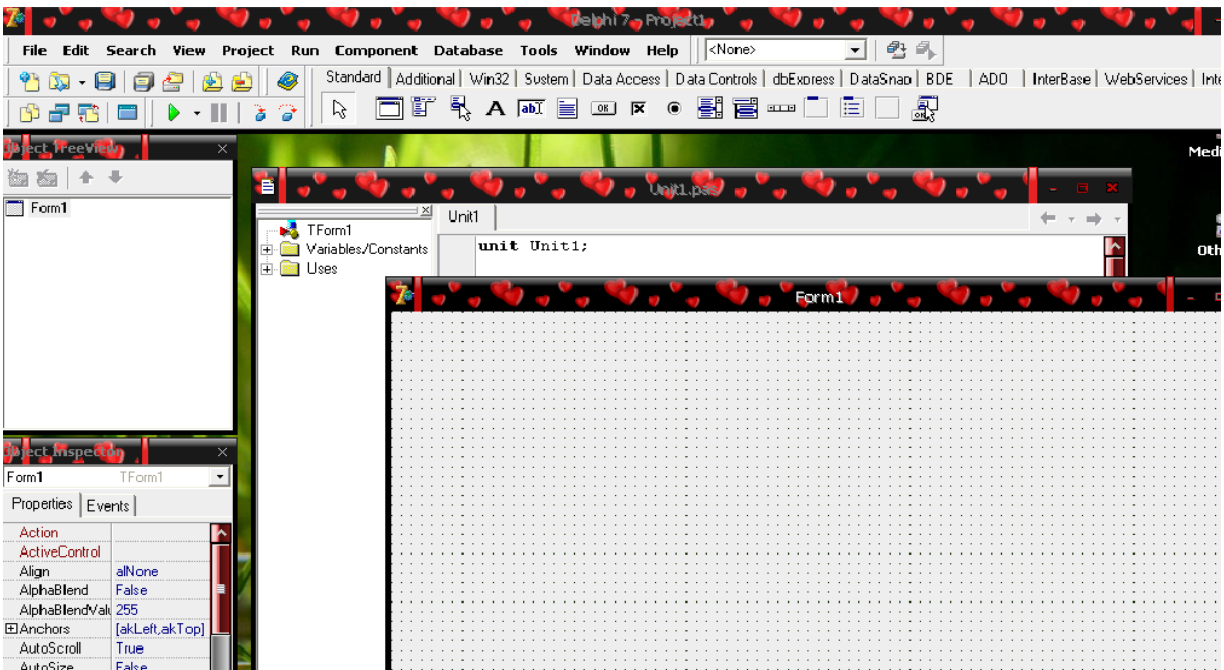
Websnap архитектураси web-иловаларни щосил =илувчи web-broker технологиясини давом эттиради ва кенгайтиради. Унинг асосида SOAP протоколи (Simple Object Access Protocol - объектга мурожаатнинг =улай протоколи) ни ишлатиш ётади.

dbExpress технологиясининг тани=ли МБ серверлари: MySQL, Oracle, DB2 каби серверларга бевосита мурожатни таъминлайди. Булар эса Windows NT Serverдан MS SQL Server га =араганда анча арзон тушади. Шунинг учун Linux системасида Delphi 6.0 да ишлаш MS корпорациясининг анча =иммат ечимларидан бирмунча арзон ёки текин былган Linux базасидаги ечимларга олиб келади. Компонентлар палитрасининг 27 та сащифасида 387 та компонент жойлашган.

DELPHI МУЩИТИ

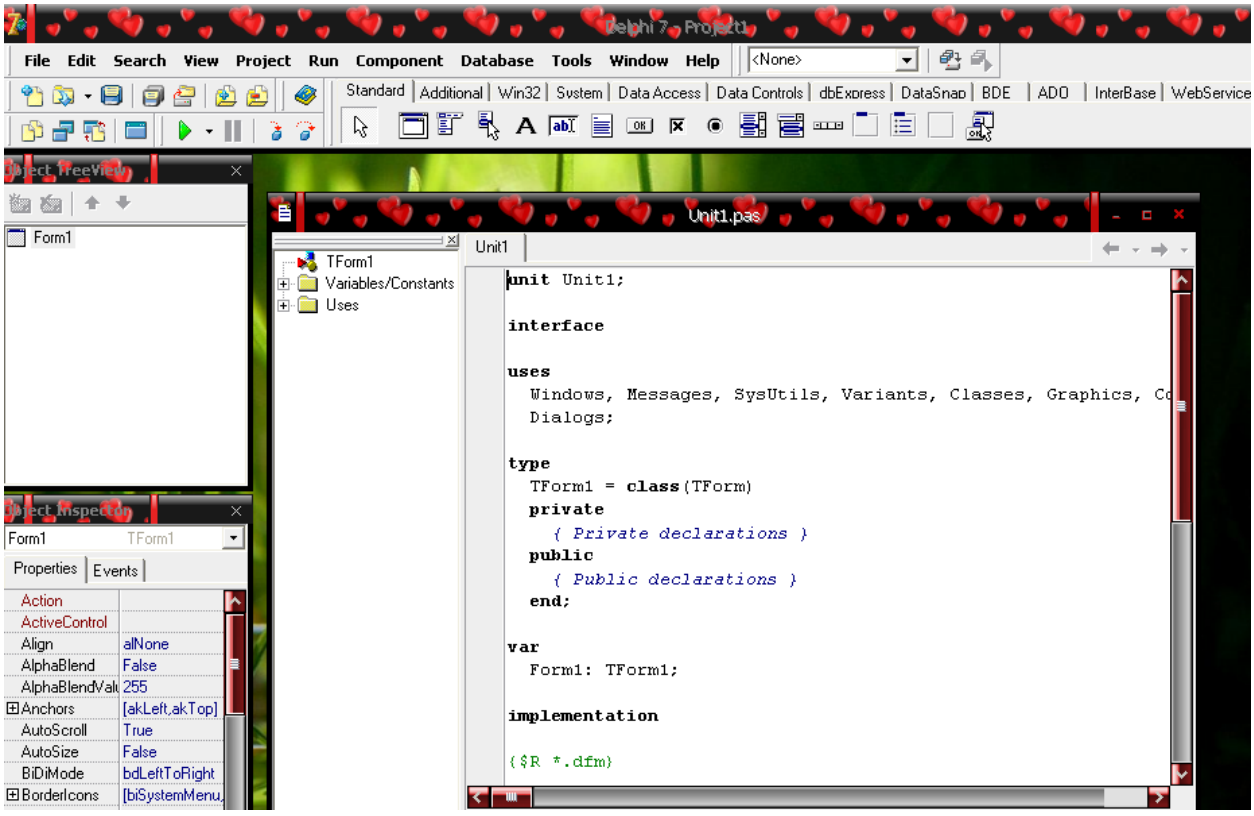
Delphi мущити бу дастурловчи ишининг ю=ори самарадорлигини таъминловчи мураккаб механизмдир. У экранда бир ва=тнинг ызида очилган бир нечта ойналарни визуал ишлатади. Щар бир ойнани керакли жойга суриш, керакли ойнага ытиш имкони мавжуд. Ҳар бир ойнанинг ани= масалаларини ечиш учун мылжалланган вазифалари мавжуд. Delphi 6.0 ни юклагач, унинг асосий 6 та ойналари экранда тасвирланади (1-расм):

1. Асосий ойна
2. Объектлар дарахти ойнаси
3. Объектлар инспектори ойнаси
4. Браузер ойнаси
5. Форма ойнаси
6. Программа коди ойнаси.



1-расм. Delphi ойналарининг умумий кўриниши.

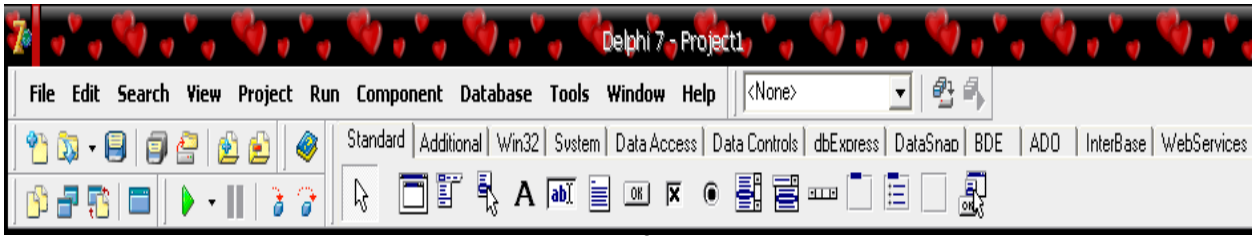
F12 клавиша босилганда форма ойнасига ёки программа коди ойнасига ытиш мумкин (2-расм).



2-расм. Код ойнасига ўтиш.

Асосий ойна

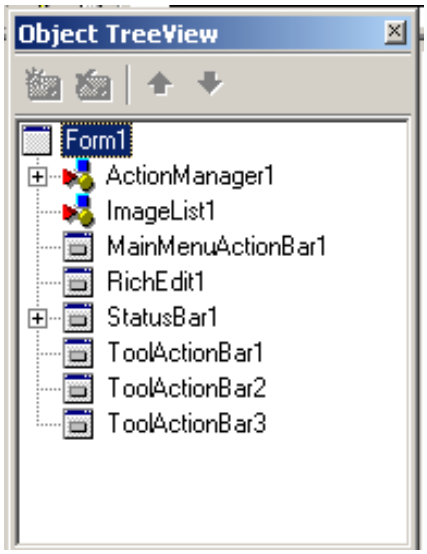
Асосий ойна тузилаётган программа лойищасини бош=аришнинг асосий функцияларини таъминлайди. Унда Delphiнинг асосий менюси, пиктографик командалар тугмачалари, компонентлар палитраси жойлашган (3-расм).



3-расм. Асосий ойнанинг кўриниши.

Объектлар дарахти ойнаси

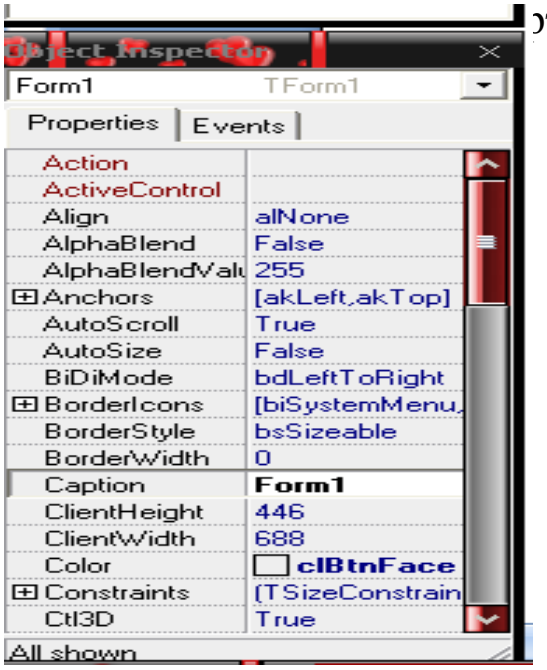
Объектлар дарахти ойнасида ишланаётган лойиҳанинг дастур модуллари тасвирланади. Унинг умумий кўриниши 4-расмда келтирилган.



4-расм. Объектлар дарахти ойнасининг кўриниши.

Объектлар инспектори ойнаси

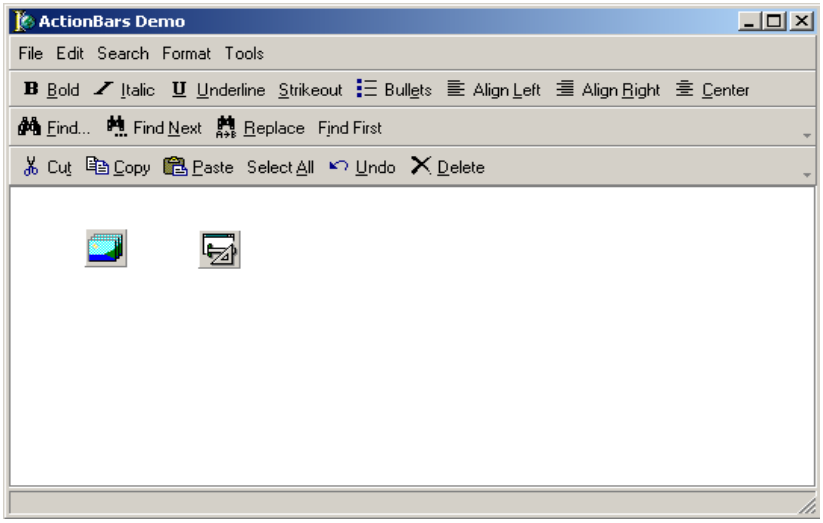
Объектлар инспектори ойнасида формага қўйилган компонентларнинг хоссалари (Properties) ҳамда ҳодисалари (Events) ҳақидаги маълумотлар жойлашган бўлади (5-расм). Бу ойнада уларнинг ...этириш имконияти мавжуд.



5-расм. Объектлар инспектори ойнасининг умумий кўриниши.

Браузер ойнаси

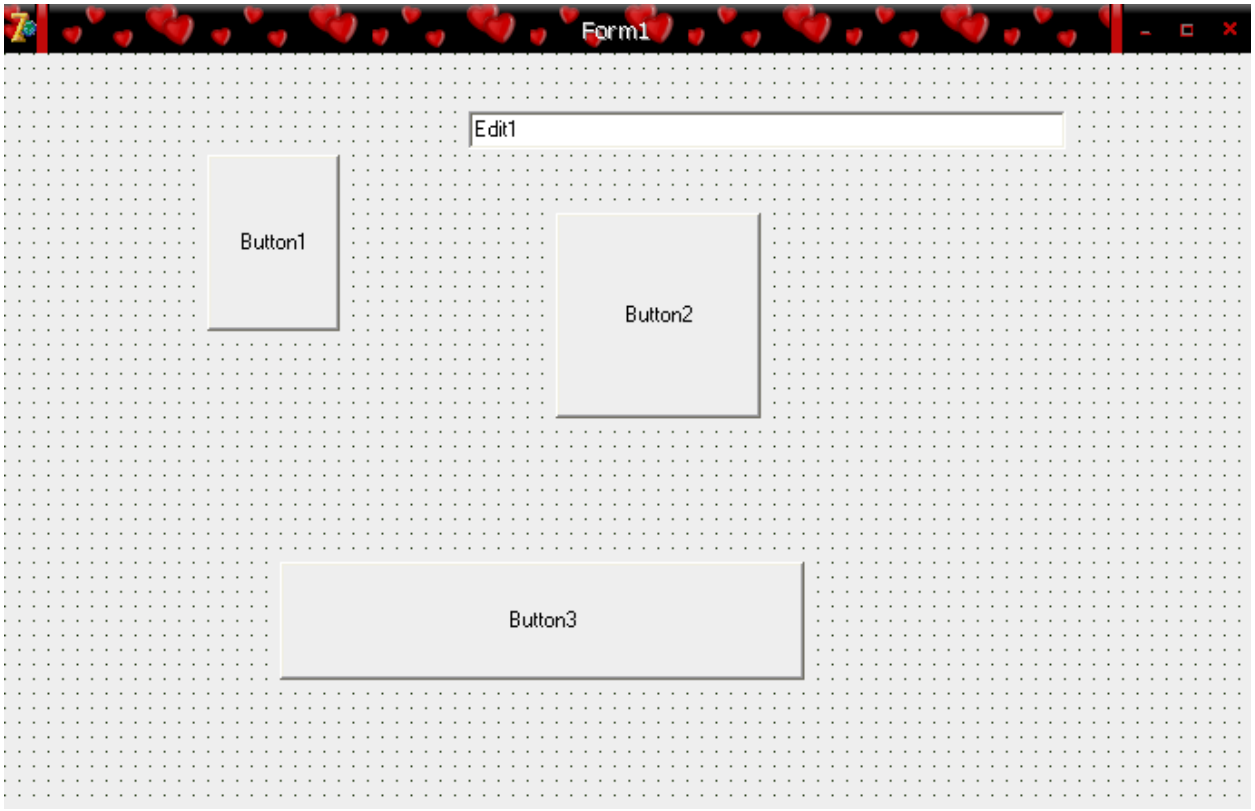
Браузер ойнасида лойиҳанинг натижаси акс эттирилади. Тузилган дастурга кўра ҳар хил кўринишга эга бўлади. Ушбу ойнанинг умумий кўриниши 6-расмда келтирилган.



6-расм. Браузер ойнасида дастур натижасининг акс этилиш лавҳаси.

Форма ойнаси

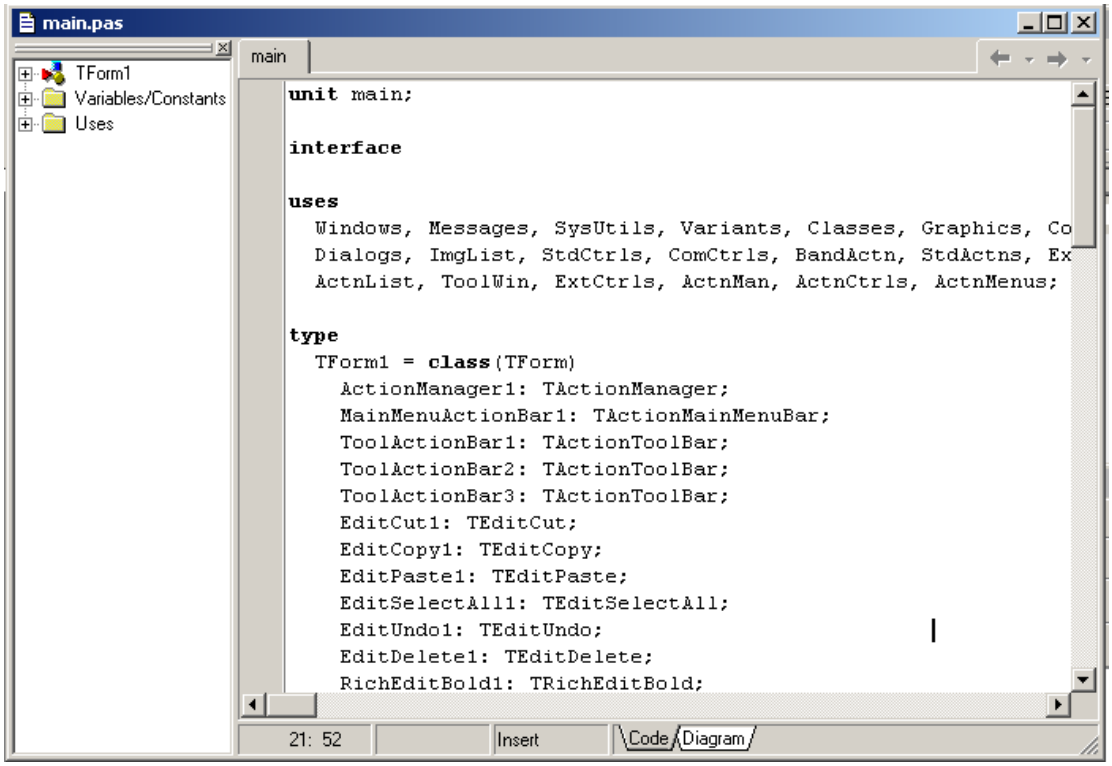
Форма ойнасига Delphiда тузилаётган дастур учун керакли бўлган компонентлар жойлаштирилади. Компонентларнинг визуал ва новизуал бўлишига кўра формада акс эттирилади (7-расм).



7-расм. Форма ойнасининг умумий кўриниши.

Программа коди ойнаси

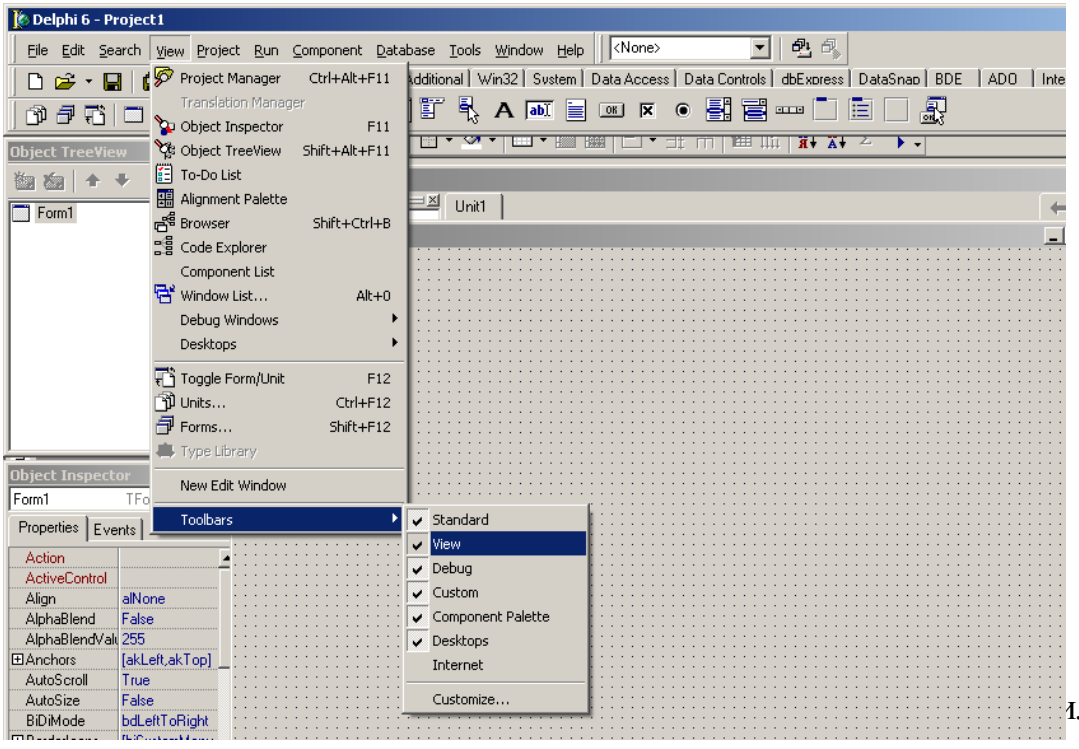
Программа коди ойнасида тузилаётган дастурнинг листинги ёзилади, яъни Object Pascal дастурлаштириш тилининг оператор ва буйруқлари ёзилиб, таҳрир қилинади. Программа коди ойнаси билан форма ойнасига F12 клавишаси орқали ўтиш мумкин. Унинг умумий кўриниши 8-расмда келтирилган.



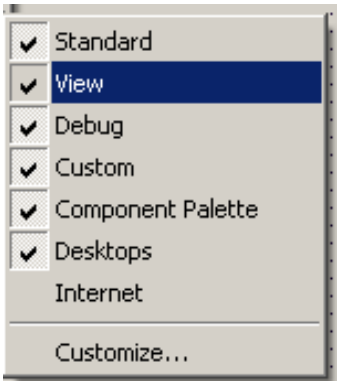
8-расм. Программа коди ойнасининг кўринишидан бир лавҳа.

ПИКТОГРАФИК ТУГМАЧАЛАР

Пиктографик тугмачалар асосий менюнинг энг асосий былимларига тезда мурожаат =илишни таъминлайди. Функционал белгисига кыра улар 7 та гурущга былинган (9-расм).

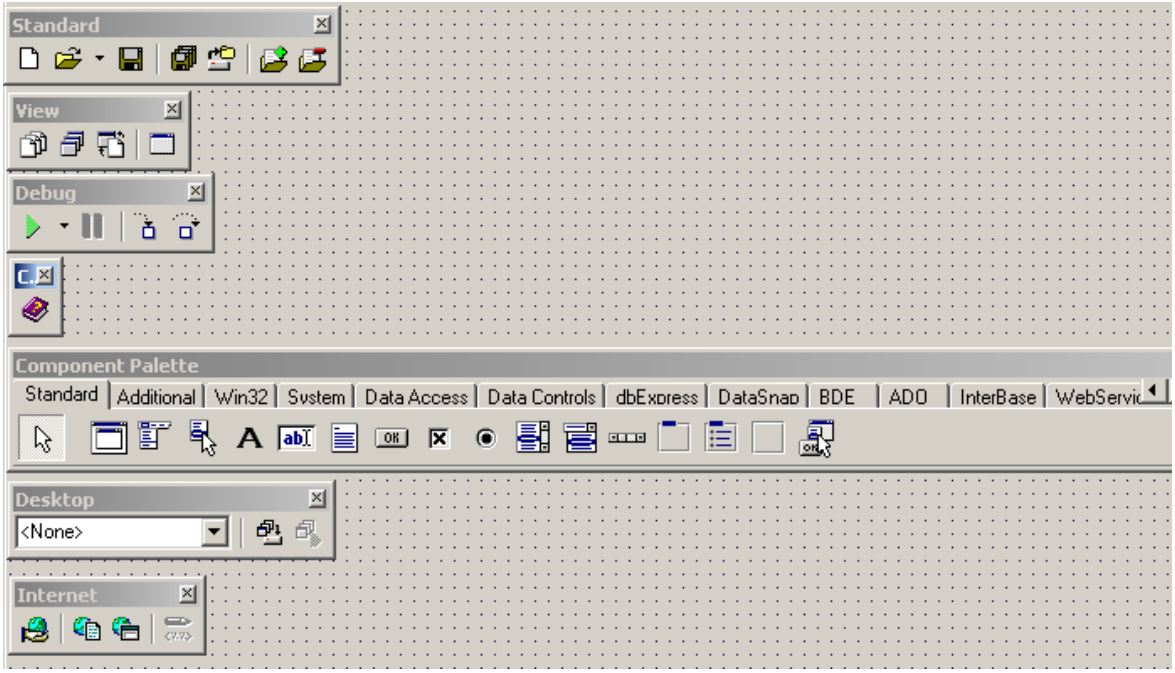


9-расм. ...илади.



Щар бир гурущ алощида панелга эга. Улар:

1. Standart гурущи. Унда 7 та тугмачаси бор.
2. View гурущи. Унда 4 та тугмачаси бор.
3. Debug гурущи. Унда 4 та тугмачаси бор.
4. Custom гурущи. Унда 1 та тугмачаси бор.
5. Component Palette гурущи. Унда компонетлар саҳифалари Delphiнинг версиялари кўра жойлаштирилган бўлади.
6. Desktops гурущи. Унда 3 та тугмачаси бор.
7. Internet гурущи. Унда 3 та тугмачаси бор.



10-расм. Пиктографик тугмачаларнинг панеллардаги кўриниши.

ОЙНАЛАРНИ ЫРНАТИШ

Delphiнинг 5-версиясидан бошлаб Desktops гурущи киритилиб, ундаги 3 та янги интерфейсли элемент билан бойитилди. Бу инструментлар ёрдамида дастурловчи Delphi нинг ойналарини жойлаштиришни бир нечта вариантларини тайёрлаб, ырнатиш файлида са=лаб =ыйиши мумкин.

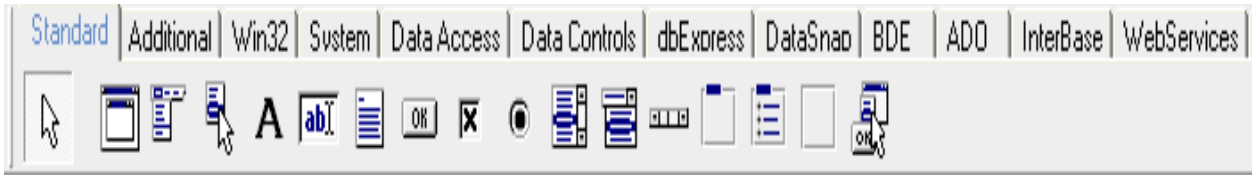
Асосан иккита ёки учта асосий ойнанинг конфигурацияси танланади: форма яратиш режими учун, кодлаштириш учун ва созлаш учун. Формани ишлаш жараёнида экранда форманинг ызи, объектлар дарахти ва объектлар инспектори кыриниб туриши керак. Бу ойналарнинг щолати ва ылчамларини ырнатиб, уларни масалан, DezignDesk номида са=лаб =ыйиш мумкин.

Кодлаш режимида ойнани иложи борича каттаро= =илиб олиш ма=садга мувофи=дир. Бундай конфигурацияни масалан, CodeDesk номида са=лаб =ыйиш мумкин.

Нищоят созлаш режимида кодлаш ойнасига айрим ойналарни, масалан, Watches ва Breakpointsни =ышиб =ыйиш мумкин. Бундай конфигурацияни DebugDesk номи билан са=лаб =ыйиш мумкин.

КОМПОНЕНТЛАР ПАЛИТРАСИ

Компонентлар палитраси Delphiнинг асосий бойлиги щисобланади. У асосий ойнанинг ынг =исмини эгаллайди щамда зарур компонентни тезда =идиришни таъминлайди. Компонент деб, =андайдир функционал элемент тушунилади. Унинг ани= хоссалари билади. Дастурловчи компонентни форма ойнасида жойлаштиради.



11-расм. Компонентлар палитрасининг Standart саҳифасининг умумий кўриниши.

ОБЪЕКТЛАР ИНСПЕКТОРИ ОЙНАСИ

Формага жойлаштириладиган ихтиёрий компонент =андайдир параметрлар йи\индисидан ташкил топади. Масалан, жойлашиши, ранги, ылчами ва бош=алар. Кыпгина параметрларни ызгартириш учун объект инспектори хизмат =илади. Бу ойнада иккита сащифа мавжуд: Properties-хоссалари, Events-щодисалари. Properties сащифаси компонентларнинг хоссаларни ызгартириш учун хизмат =илади.

Events сащифасида эса у ёки бу щодисага былган компонентнинг реакциясини ани=лаш учун хизмат =илади.

ВИЗУАЛ ДАСТУРЛАШ АСОСЛАРИ

Янги лойища очилган защоти кодлаш ойнасида =уйидаги =аторлар пайдо былади:

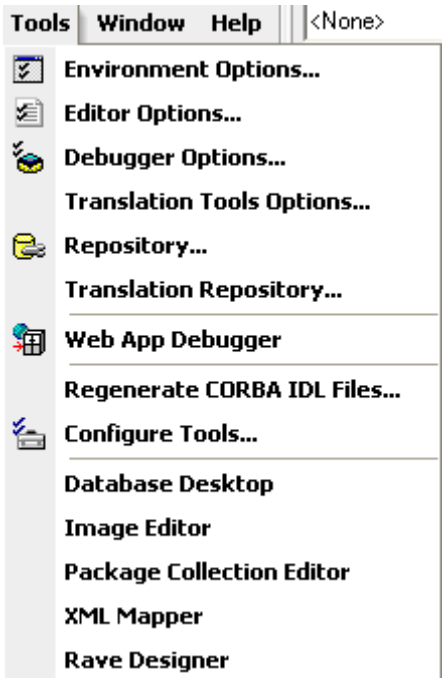
```
unit Unit1;

interface

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

type
    Tform1 = class(Tform)
     private
         {Private declarations}
       public
          {Public declaratoins}
       end;

var
       Form1: Tform1;

implementation

   {$R *.DFM}

end.
```

+уйидагиларга эътибор беришингиз лозим:

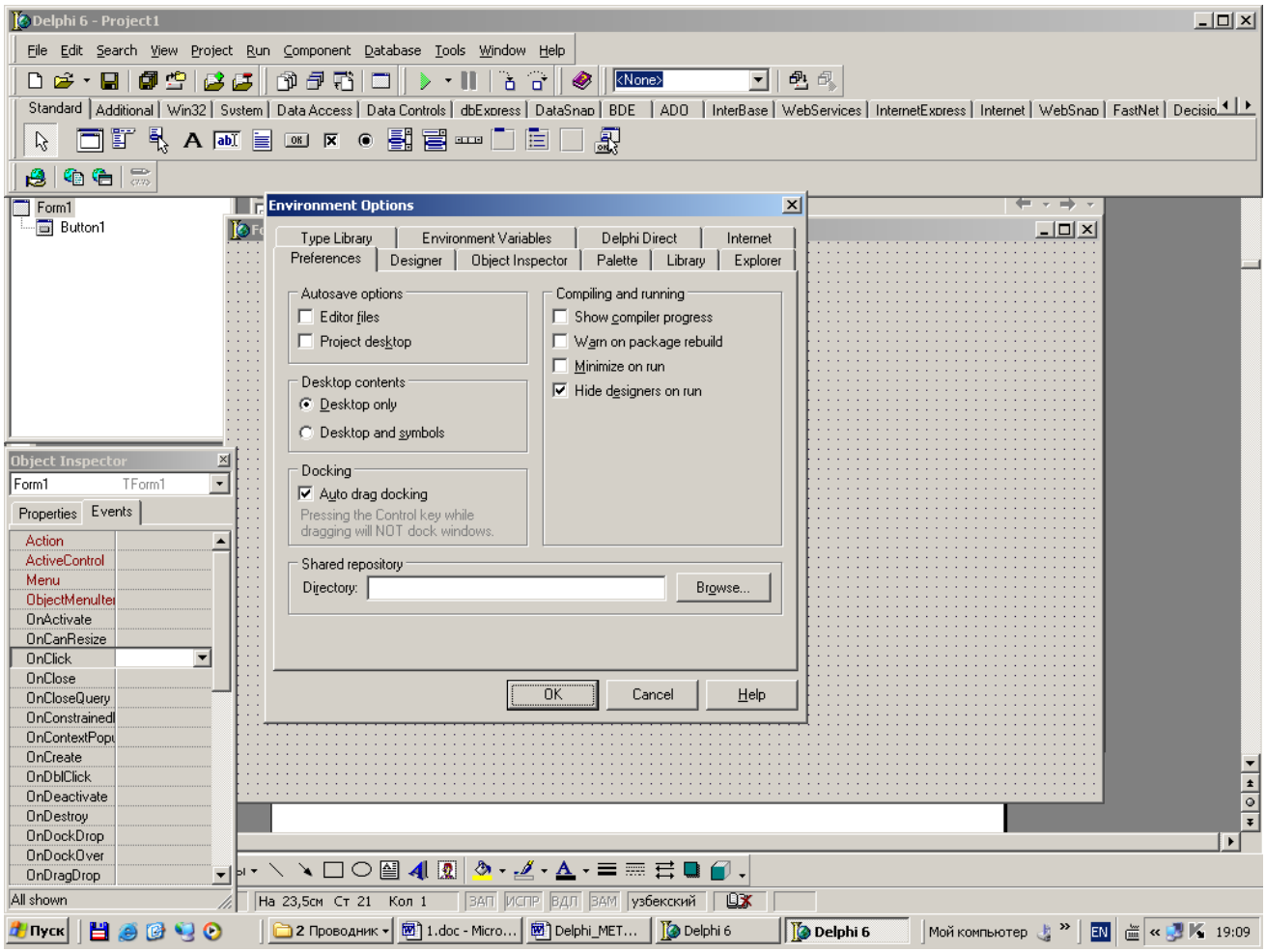
1) Дастлаб программани ишлатишдан олдин албатта ызингизга янги папка очишингиз ма=садга мувофи= былади. Шу папкага щосил =илаётган программани ёзиб борасиз.

2) Сиз щосил =илаётган программанинг охирги версиясини Delphi мущити автоматик равишда са=лаб =ыйиши учун Delphiнинг стандарт мущитига бироз ызгартириш киритиш керак былади.

Бунинг учун Tools/Environment Options танланади:



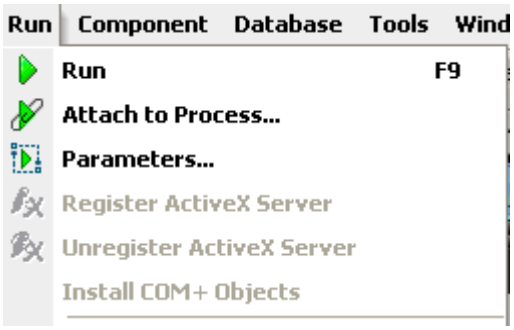
Бу ерда Preferences сащифаси фаоллигига =аралади. Бу сащифанинг ю=ориги чап бурчагининг AutoSaveOptions гурущида Editor Files ва Desktop ёзувлар мавжуд (12-расм). Бу ёзувлар танланса, программа кодини ва Delphi ойналарининг умумий кьринишини автоматик равишда са=лайди.



12-расм. Environment Options нинг Preferences саҳифасининг кўриниши.

Компиляция жараёнини текшириб бориш учун Compiling and Running гурущидаги Show Compiler Progress ёзувини танлаш керак былади.

Программани ишлатиш учун F9 клавишаси ёки ▶ тугмачаси босилади:



Шу усулда Delphi программа 3 та асосий этапдан ытади: компиляция, компоновка ва бажарилиш.

DELPHIДА ИШЛАТИЛАДИГАН ИСМЛАР

Delphiда фа=ат лотин щарфлари, сонлар ва тагига чизилиш белгисидан фойдаланиб исмлар ёзилади. Исм сондан бошланмаслиги керак.

Delphiнинг 1-версиясида исмлар узунлиги 8 та белгидан ошмаслиги керак. Кейингиларида эса кыпро= символлардан фойдаланиш мумкин.

Щар бир щосил =илинадиган форма учун модуль щосил =илинади. Программани компиляциялаш жараёнида Delphi щар бир модуль учун .pas, .dfm, .dcu кенгайтмали файлларни щосил =илади.

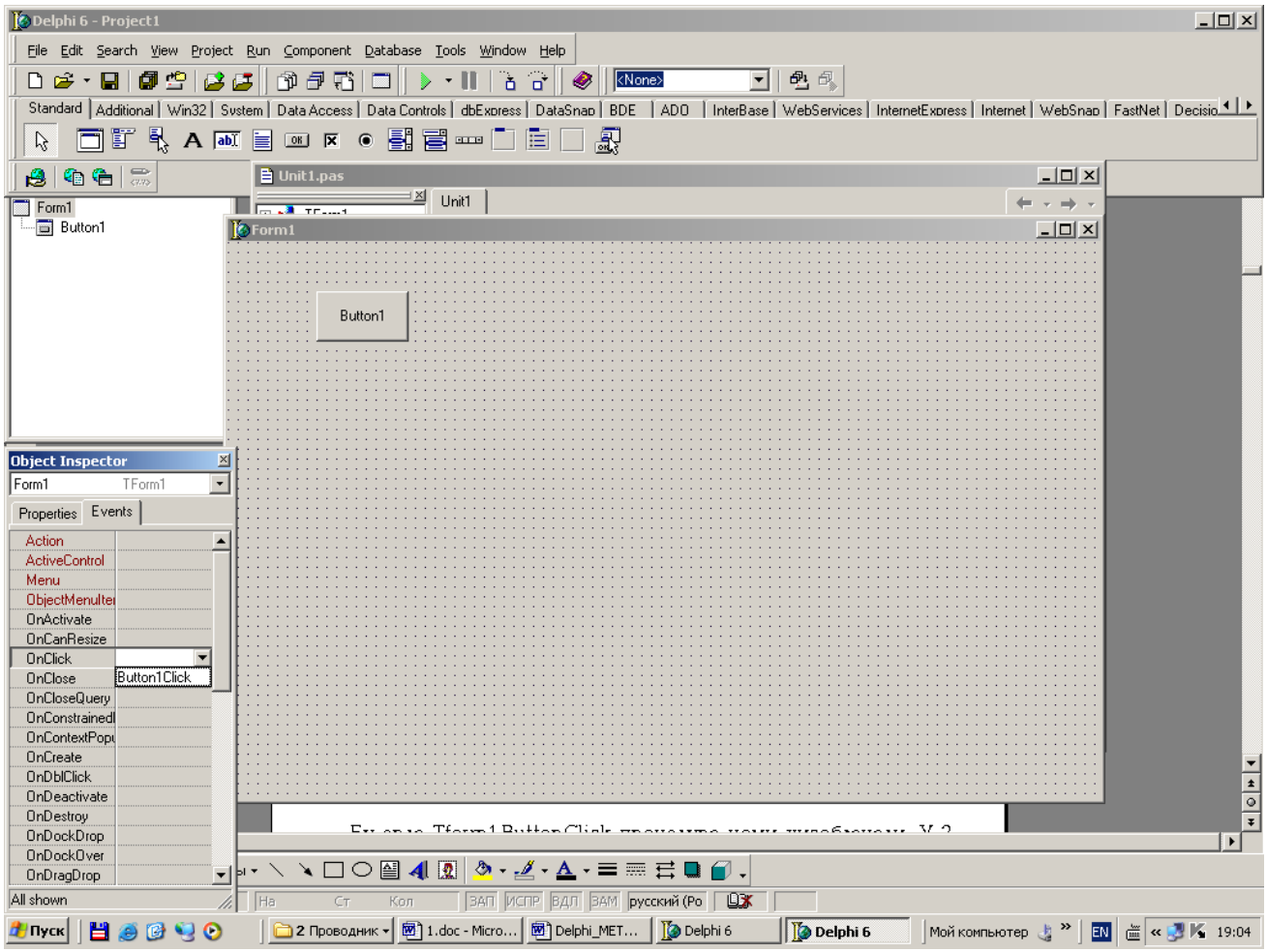
.dcu файллар компиляторлар ор=али щосил =илинади. Кейинчалик бу файлдан фойдаланиб, компоновкаловчи .exe тоифали файл щосил =илади.

Щодисаларга былган ижро

Бунинг учун формага Button компонентни ырнатамиз ва бу компонентни щодисаларга былган ижросини кыриб чи=амиз.

OnClick - щодисаларни =айта ишлагич

Ишлаётган программада сич=ончанинг тугмачаси босилганда OnClick щодисаси пайдо былади (13-расм). Дастлаб тугмача босилганда щеч =андай ижро былмайди. Ижрони щосил =илиш учун Object Pascal тилида программа былагини ёзиш керак былади. Уни щодисаларга ижро деб аталади. Object Pascal тилида программа былаги бу махсус =исмпрограмма былиб, бош=ача =илиб айтганда, уни процедура дейиш щам мумкин. У процедура кыринишида ташкиллаштирилади.

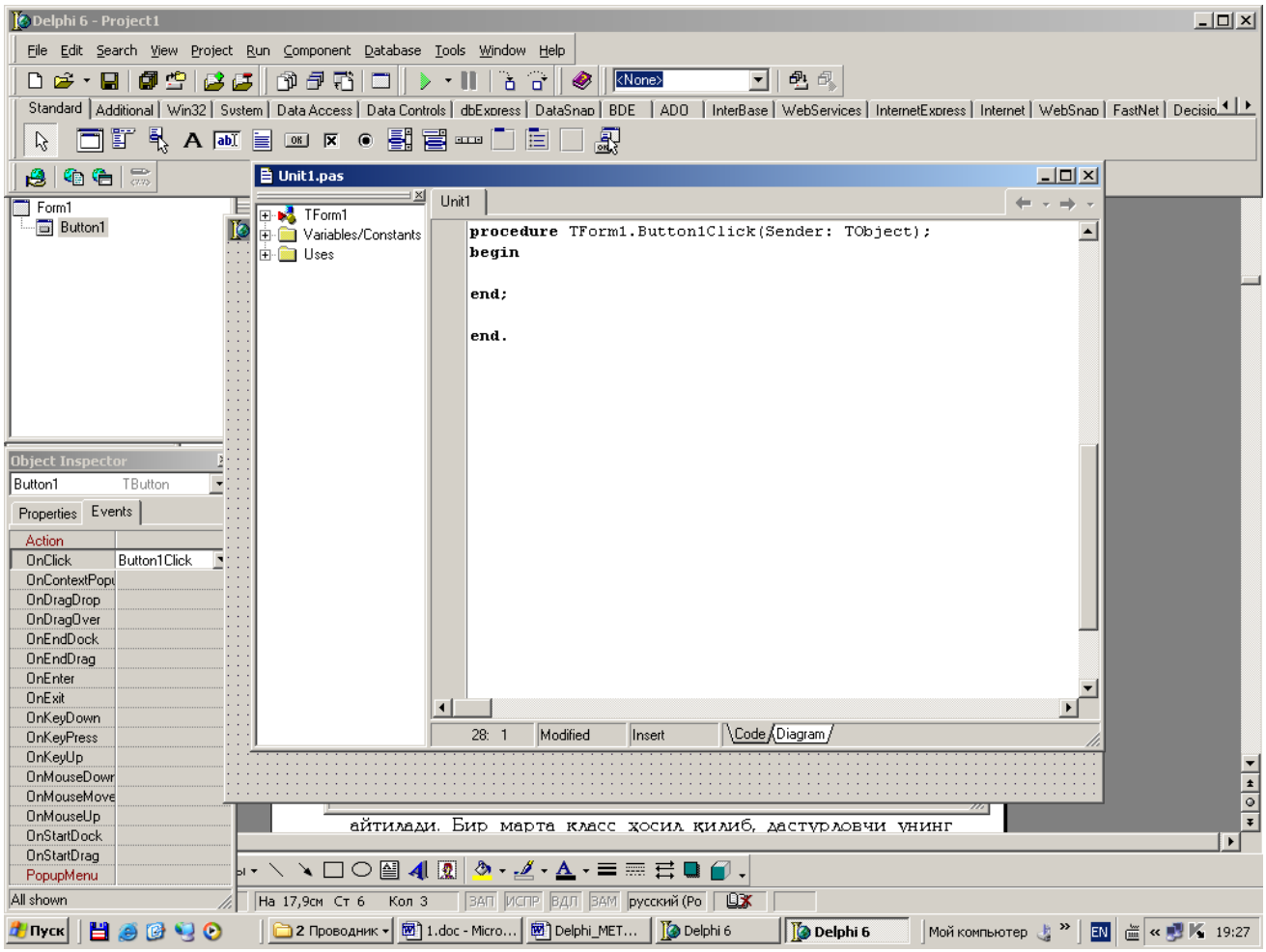


13-расм. OnClick щодисасини чақириш.

Компонентга сич=онча кетма-кет 2 марта босилса, код ойнаси очилади ва =уйидаги матн щосил былади (14-расм):

```
Prosedure Tform1.ButtonClick (Sender:TObject);
Begin
    .....
end;
```

Бу ерда Tform1.ButtonClick процедура номи щисобланади. У 2 та =исмдан иборат: класс номи - Tform1 ва процедура номи - ButtonClick.



14-расм. Компонентга сич=онча кетма-кет 2 марта босилганда очилган код ойнасининг кўриниши.

Класслар деб функционал тугалланган программа былакларига айтилади. Бир марта класс щосил =илиб, дастурловчи унинг нусхаларини бош=а ёки шу программага =ышиб =ыйиши мумкин. Бу программанинг ю=ори мащсулдорлигига олиб келади. Щар бир компонент ани= бир классга тааллу=ли былади. Формага =ыйиладиган компонентлар эса сонли индекс =ышилган класс номини олади. Delphiда барча класслар “Т” щарфи билан бошланади. Демак Tform1 класс номини англатиб, стандарт класс Tform намунаси быйича щосил =илинган.

Программанинг бошида

```
Type
    Tform1 = class(TForm)
```

```
        Button1 :  Tbutton;
        Label1 : Tlabel;
        Procedure ButtonClick(Sender: TObject);
    private
      {Private declarations}
    public
       {Public declaratoins}
    end;

var
    Form1 : Tform1;
```

келтирилганлигини кыриш мумкин.

Бу ерда Tform1 = class(TForm) дегани янги Tform1 классини стандарт класс – Tform дан келиб чи==анини билдиради.

```
    form1 : Tform1;
```

эса шу класснинг нусхасини form1 ном билан щосил =илади.

Формага =ыйилган button, label компоненти эса программада =уйидаги кыринишда былади:

```
    Button1 : Tbutton;
    Label1: Tlabel;
```

Button1 компоненти стандарт класс - TButton нинг нусхаси эканлигини билдиради.

```
    Sender : Tobject;
```

бу дегани Sender номли параметр Tobject классига тааллу=ли.
Масалан, Button1 тугмачаси босилганда, унга ижро былиши учун шу тугмачага тез икки марта сич=ончанинг чап тугмачаси босилса, шу процедуранинг танаси щосил былади. ButtonClick процедурасининг танаси begin ва end орасида ёзилади. Бу ерда масалан,

MessageBeep(MB_OK);

ёки

Close;

деб ёзиш мумкин. Программа ишлатилганда Button1 тугмачаси босилса, биринчи щолда товуш чи=ади, иккинчи щолда эса процедурадан чи=иб кетилади.

OBJECT PASCAL ТИЛИГА КИРИШ

Pascal тилида программа лотин алифбосидаги щарфлардан (катта кичик), сонлардан, тагига чизилиш белгисидан ва стандарт тиниш белгиларидан фойдаланган щолда ёзилади.

Программада калит сызлари (стандарт ва фойдаланувчи ани=лайдиган) щамда идентификатор ва ифодалар ишлатилади. Идентификатор сифатида щарф, сон ва тагига чизилиш белгисининг ихтиёрий кетма-кетлигидан фойдаланиш мумкин. Фа=ат сондан бошланмаслиги шарт. Масалан, Unit1, integer, x, for, There_are, Go98.

Фойдаланувчининг идентификатори сифатида калит сызлар ва стандарт идентификаторларни ишлатиш мумкин эмас.

Программа элементлари бу компилятор учун ани= =имматга эга былган, унинг минимал ажралмас =исмидир. Унга =уйидагилар киради:

1. Калит сызлар;
2. Идентификаторлар;
3. Тоифалар;
4. Константалар;
5. Ызгарувчилар;
6. Белгилар;
7. +исм программалар;
8. Изощлар;

АМАЛЛАР

Амалларни амал белгиларидан ва =авслардан фойдаланган щолда ифодалаш мумкин: +, -, *, /, ^, (,), SQR, SQRT.

Object Pascalда =уйидаги амаллар ани=ланган:
1. Унар: Not, @.
2. Мультипликатив: *, /, div, mod, and, shl, shr.
3. Аддитив: +, -, or, xor.
4. Муносабат: =, <>, <, >, <=, >=, in.

div – былишдан =олган бутун =исми.
mod – былишдан =олган =олди= =исми.
Shl - чапга суради
Shr - ынгга суради.

Бутун сонлар билан ишлаганда:
Ord(x) – x ни ызини =айтаради: Ord(x)=x.

Pred(x) – x дан битта олдингисини =айтаради: Ord(Pred(x))=Ord(x)-1.

Succ(x) - x дан битта кейингисини =айтаради: Ord(Succ(x))=Ord(x)+1.

Манти=ий сонлар билан ишлаганда:
Ord(False)=0
Ord(True)<>0
Succ(False)=True
Pred(True)=False.

ТОИФАВИЙ КОНСТАНТАЛАР

Object Pascalда тоифавий константалардан фойдаланиш мумкин. Унинг умумий кыриниши =уйидагича:

<идентификатор> : <тоифа> = <=иймат>;

Бу ерда <идентификатор> - константанинг идентификатори;

<тоифа> - константанинг тоифаси; <=иймат> - константанинг =иймати.

Delphi 6 дан бош=а версияларида программани ичида тоифавий константаларга бош=а =иймат бериш мумкин.

Тоифавий константалар вариант, файл, объект ва класслардан бош=а ихтиёрий тоифада былиши мумкин. Масалан,

```
Type
    Colors = (white, red, black);
Const
    CurrCol : colors = red;
    Name : string = ‘Adham’;
    Year : Word = 1989;
    X : real = 0.1;
    Min  : Integer = 0;
    Max  : Integer = 0;
    Days   : 1..31 = 1;
    Answer  : char = ‘Y’;
    Digit : array[0..9] of char = (‘0’,’1’,’2’,’3’,’4’,’5’,’6’,’7’,’8’,’9’,’10’);
    Digit1 : array[0..9] of char = (‘0123456789’);
```

МАЪЛУМОТЛАР ТОИФАЛАРИ

Маълумотлар тоифалари =уйидагиларга былинади: бутун сонлар, ща=и=ий сонлар, манти=ий тоифалар, матнавий тоифалар, =атор тоифалар, саналувчи тоифалар, ва=т-сана тоифаси. +уйида санаб ытилган тоифаларни бирма-бир кыриб чи=амиз:

1. Бутун сонлар =уйидаги жадвалда келтирилган тоифаларга былинади:

№	Номи	+иймати
1.	integer	-2 147 483 648 ¼ + 2 147 483 647
2.	cardinal	0 ¼ 4 294 967 295
3.	shortint	-128 ¼ +127

4.	smallint	-32 768 ¼ +32 767
5.	longint	-2 147 483 648 ¼ + 2 147 483 647
6.	int64	-2^{63} ¼ $+2^{63}-1$
7.	byte	0 ¼ +255
8.	word	0 ¼65 535
9.	longword	0 ¼4 294 967 295

2. Ща=и=ий сонлар =уйидаги жадвалда келтирилган тоифаларга былинади:

№	Номи	Узунлиги (байтда)	+иймати
1.	Real	8 байт	$5,0*10^{-324}$¼$1,7*10^{308}$
2.	Single	4 байт	$1,5*10^{-45}$¼$3,4*10^{38}$
3.	Double	8 байт	$5,0*10^{-324}$¼$1,7*10^{308}$
4.	Extended	10 байт	$3,4*10^{-4951}$¼$1,1*10^{4932}$
5.	Comp	8 байт	-2^{63}¼ $+2^{63}-1$
6.	Currency	8 байт	±922337203685477,5807

3. Манти=ий тоифалар =уйидаги жадвалда келтирилган тоифаларга былинади:

№	Номи	Узунлиги (байтда)
1.	Boolean	1 байт
2.	Bytebool	1 байт
3.	Bool	2 байт
4.	Wordbool	2 байт
5.	Longbool	4 байт

Бу тоифаларнинг =иймати сифатида фа=атгина true ва false =ийматлар ишлатилади.

4. Символли (матнавий) тоифалар. Бу тоифаларнинг =иймати сифатида ШЭЩМнинг барча символлари келиши мумкин. Щар бир символга 0¼255 орали=да былган сон ёзилган былади. Матнавий тоифаларнинг номланиши =уйида келтирилган:

char – ихтиёрий символ былиши мумкин.

chr(B) - B ифодани (байт типидаги) символга айлантириб беради. Уни ызини =иймат сифатида =абул =илиб олади.

Upcase(ch) - лотинча кичик щарфларни катта лотин щарфларига ытказиб беради.

5. +атор тоифалар. Бу тоифаларнинг номланиши =уйидагича: shortstring, string[n] - n ни =иймати сифатида 0¼255 та символ келиши мумкин.
String (узун =атор), Widestring (кенг =атор) - 0¼2 Гбайт узунликда былиши мумкин.
Pchar (нул терминал =атор) - 0 ¼2 Гбайт (2^{31}) узунликда былиши мумкин.

6. Саналувчи тоифалар. Буларга кетма-кет саналувчи бутун сонли =ийматларни киритиш мумкин. Бу тоифаларнинг максимал =иймати 65536 =ийматга тенг.

7. Сана-ва=т тоифаси. Бу тоифа TDateTime стандарт идентификатор билан ани=ланади. У бир ва=тнинг ызида санани ва ва=тни са=лаш учун хизмат =илади. Ички тузилиш быйича у 8 байтни эгаллайди ва Currency тоифасига ыхшаб бутун =исмида сана, вергулдан кейинги =исмда ва=т щисобланади. –693594 дан кичик =ийматларни =абул =илмайди, чунки бу сон мелодий йилнинг бошланишига ты\ри келади (00,00,0000 – кун, ой, йил).

Масалан: 36444,837 =иймати 11.10.1999. санага ва 20:05 ва=тга ты\ри келади.

СТРУКТУРАЛАШТИРИЛГАН ТОИФАЛАР

Object Pascal тилида 4 та турдаги структуралаштирилган тоифалардан фойдаланиш мумкин. Булар: массивлар, ёзувлар, тыпламлар ва файллар. Шу тип элементларини ташкил этувчиларнинг кыплигидан ихтиёрий структура тоифалари келиб чи=ади.

Object Pascal тилида тоифаларнинг ичида яна тоифалар щам ишлатилиши мумкин, лекин уларнинг ички чу=урлиги 2 Гбайтдан ошмаслиги керак.

МАССИВЛАР

Массивлар бош=а тилдаги массивларга ыхшайди. Унинг щар бир компоненти битта тоифага тегишли былади.

Массив тоифасининг умумий ёзилиши =уйидагича:

<массив_номи> = array [<инд_тоиф_ рыйхати>] of <тоифа>

Бу ерда <массив_номи> - ты\ри идентификатор; array, of – калит сызлар; <тоифа> - массивнинг тоифаси; <инд_тоиф_ рыйхати> - бир ёки бир нечта индексли тоифалар рыйхати.

Индексли тоифалар рыйхати сифатида Object Pascalда ихтиёрий саналувчи типлардан фойдаланиш мумкин, фа=ат 2 Гбайтдан ошмаслиги керак (яъни LongWord ва Int64 дан таш=ари).

Программада ызгарувчини массив сифатида тавсифлаш учун ты\ридан-ты\ри var былимида щам келтириш мумкин.

Масалан,

```
Var
   A, b : array [1..10] of char;
```

Агар type былимида келтирилган былса, у щолда:

```
Type
    Massiv = array[1..10] of char;
  Var
     A, b : massiv;
```

деб ёзилади.

Массив ичида массив келса, у щолда:

```
Type
  Massiv =array[1..10] of array[-2..2] of array[0..5] of char;
```

Худди шу ёзувни ихчам =илиб, =уйидаги кыринишда щам ёзиш мумкин:

```
Type
   Massiv =array [1..10, -2..2, 0..5] of char;
```

Бу кыринишдаги массивлар статик массивлар дейилади.

ДИНАМИК МАССИВЛАР

4-версиясидан бошлаб биринчи марта динамик массивлар киритилди. Бундай массив эълон =илинганда, уларнинг индекслари чегарасини кырсатиш керак эмас. Масалан,

```
A : array of integer;
B : array of array of char;
C : array of array of arrray of char;
```

Бу ерда А-динамик массив бир ылчамга, В динамик массив 2 ылчамга, С динамик массив 3 ылчамга эга.

Динамик массивнинг индексли чегарасини ани=лаш программанинг бажарилиши жараёнида SetLength функцияси ёрдамида массив инициация =илиш йыли билан амалга оширилади. Масалан,

```
SetLength(A,3);
```

Бу оператор бажарилгач, А-динамик массивга хотирадан 3 та бутун соннинг =ийматини си\адиган жой ажратилади. Массивнинг =уйи чегараси 0, ю=ори чегараси 2 былади. Динамик массивлар ыз ишини тугатгач, хотирадан жойни бышатиб =ыйиши керак былади. Масалан,

```
 SetLength(A,10);
 SetLength(B,20);
// массивлар ишлатилади.
```

…
// хотирани бышатамиз:

```
A := nil;
Finalize (B);
```

Оддий массив ылчамлари бир хил былиши керак эди. Масалан, 2x3 массивда 2 та =аторда 3 тадан устунли былади. Динамик массивда эса щар бир ылчамда щар хил узунликдаги ылчамли массивлар ишлатилиши мумкин. Масалан,

```
SetLength(A,3);
```

// Щар бир устуннинг узунлиги турлича:

```
Setlegth(a[0],3);
Setlegth(a[1],4);
Setlegth(a[2],5);
```

ЁЗУВЛАР

Ёзув бу маълумотлар структураси былиб, у ёзувлар майдонлари деб номланувчи компонентлардан ташкил топади. Ёзувлар элементлари (майдонлари) щар хил тоифага тегишли былиши мумкин. Унинг умумий кыриниши =уйидагича:

<ёзув_номи > = record <майдон_ рыйхати> end;

Масалан,

```
 type
       birthday = record
          My_day : byte;
          My_year : word
      End;
 Var
     A, B : birthday;
```

Бу ёзувларга =уйидаги кыринишда мурожаат =илиш мумкин:

```
A.My_day:=27;
B.My_year:=1985;
```

Ичма-ич ёзувлар =уйидаги кыринишда ташкил этилади:

```
Type
   Birthday = record
     My_day : byte;
     My_year : word;
   End;
Var
   A : record
     Name : string;
        Bd : Birthday;
   End;
```

Бу кыринишдаги ёзувларга мурожаат =илиш учун =уйидаги командани бериш мумкин:

```
Begin
    . . .
      If a.Bd.My_year = 1995 then ...
      . . .
End;
```

Ёзув майдонларига мурожаатни осонлаштириш with оператори ишлатилади. Унинг умумий кыриниши =уйидагича:

With <ызгарувчи> do <оператор>;

Масалан,

```
      a.Name:= ‘Asror’;
      a.Bd.My_year:= 2002;
      a.Bd.My_day:= 20;
```

деб ёзишни ырнига =уйидагича ёзиш мумкин:

```
with a do
        Begin
          Name:= ‘Asror’;
        Bd.My_year:= 2002;
          Bd.My_day:= 20;
        End;
```

ТЫПЛАМЛАР

Тыпламлар бу бир-бири билан манти=ан бо\ланган бир тоифага тегишли объектларнинг кыплигидир.

Тыпламга кирувчи элементларнинг сони 0¼255 гача былиши мумкин. Массив ва ёзувлардан фар=ли равишда тыпламлар ундаги элементлар сонининг ызгарувчанлиги билан фар= =илади.

Иккита тыплам фа=ат ва фа=ат шу щолда эквивалент щисобланади, агар уларнинг щар бир элементлари бир хил былса, лекин уларнинг келиш тартиби ащамиятли эмас. Умумий кыриниши =уйидагича:

<тыплам_номи> = set of <база_тоифаси>

<База_тоифаси> сифатида word, integer, longint , int64 дан бош=а ихтиёрий саналувчи тоифалардан фойдаланиш мумкин. Масалан,

```
Type
        Digitchar = set of ’0’..’9’;
        Digit       = set of 0..9;
Var
        S1, s2, s3 : Digitchar;
        S4, s5, s6 : Digit;
Begin
        . . .
        S1 := [‘1’,’2’,’3’];
        S2 := [‘3’,’2’,’1’];
        S3 := [‘2’,’3’];
        S4 := [0..3,6];
```

```
    S5:= [4,5];
    S6:= [3..9];
    . . .
End;
```

Бу мисолда S1 ва S2 тыпламлар эквивалент. S3 тыплам эса S1 ва S2 тыплам ичига киради.

Тыпламлар устида =уйидаги амалларни бажариш мумкин:

1) * - тыпламларнинг кесишиши. Масалан,

   s4*s6 = [3]; S4*S5 = быш.

2) + - тыпламларнинг бирлашиши. Масалан,

   s4+s5 = [0,1,2,3,4,5,6];
   s6+s5 = [3,4,5,6,7,8,9];

3) - - тыпламларнинг айирмаси. Масалан,

   s6-s5 = [3,6,7,8,9];
   s4-s5 = [0,1,2,3,6];

4) = - эквивалентликка текшириш. Натижаси true, агар эквивалент былса.
5) <> - ноэквивалентликка текшириш. Натижаси true, агар ноэквивалент былса.
6) <= - ичига киришини текшириш. True, агар 1-тыплам 2-тыпламга кирса.
7) => - ичига киришини текшириш. True, агар 2-тыплам 1-тыпламга кирса.
8) in – тыпламга тегишлилигини текшириш. Бунда 1-элемент ифода, 2-элемент тыплам былиши керак. Масалан,

3 in S6 натижаси true
2*2 in S1 натижаси false.

9) include (s, i)- тыпламга янги элемент =ышади. Бу ерда S – тыплам, i – тыпламга =ышилиши керак былган TsetBase тоифасининг элементи.
10) exclude(s,i)- тыпламдан элементни олиб ташлаш.

КЫРСАТГИЧЛАР ВА ДИНАМИК ХОТИРА

Динамик хотира

Динамик хотира бу ШЭЩМнинг оператив хотираси былиб, программа ишлаётганда хизмат =илади. Маълумотларни динамик жойлаштирганда, бу маълумотларнинг ми=дори щам, тоифаси щам олдиндан ани=ланмаган былади.

Кырсатгичлар

Кырсатгич бу ызгарувчи былиб, унинг =иймати сифатида хотира байтининг адреси келади.

Кырсатгичлар ёрдамида Object Pascalдаги ихтиёрий тоифаларни динамик хотирада жойлаштириш мумкин.

Кырсатгичларни =уйидагича ёзиш мумкин:

```
Var
   P1 :^integer;
   P2 :^real;
  Type
    PersonPointer = ^PersonRecord;
    PersonRecord = record
            Name : string;
              Job : string;
             Next : PersonPointer
    end;
```

Буни тоифалаштирилган кырсатгичлар дейилади. +андайдир ани= тоифа билан бо\ламасдан щам ёзиш мумкин. Масалан,

```
Var
```

```
P : pointer;
```

Буни тоифалаштирилмаган кырсаткичлар дейилади. Масалан,

```
Var
   P1, P : ^integer;
   P2 : ^real;
   P3 : pointer;
```

былса, у щолда

```
Begin
  . . .
   P1 := P;
  . . .
 end
```

деб ёзиш мумкин. Лекин,

```
Begin
  . . .
   P1 := P2;
  . . .
end;
```

деб былмайди.

Тоифалаштирилмаган кырсатгичлар учун эса =уйидагича ёзиш мумкин:

```
P3 := P2;
P1 := P3;
```

Динамик хотирани ажратиш ва бышатиш

Ихтиёрий динамик ызгарувчи учун ажратиладиган хотира NEW процедураси ёрдамида амалга оширилади. Масалан,

```
Var
  P3, P1 : ^integer;
      P2 : ^real;
Begin
   New (P3);
   New (P2);
   . . .
end;
```

Кейин эса

```
P3^ := 2; // P3 ни хотира майдонига 2 сонини ёзади.
P2^ := 2 * pi; // P2 ни хотира майдонига 6.28 сонини ёзади.
```

Dispose(P3); - олдин олинган хотирани =айтадан бериш процедураси. Масалан,

```
Const
Pr :^real=nil;
Begin
  . . . .
  if pr = nil then new(pr);
  . . . .
   Dispose(pr);
   Pr := nil;
   . . . .
end;
```

Булар тоифалаштирилган кырсаткичлар учун эди. Тоифалаштирилмаган кырсаткичлар учун эса =уйидаги процедуралар билан ишлаш имконияти мавжуд:

```
GetMem(p, size); - хотирадан жой олиш.
FreeMem(p,size); - хотирани бышатиш.
```

бу ерда p-тоифалаштирилмаган кырсаткич, size - бышатилиши керак былган хотиранинг байтлардаги ылчами.

Тоифалаштирилмаган кырсаткичлар билан ишлаганда жуда эщтиёткор былиш лозим. +анча хотира олинган былса, худди шунчасини бышатиш керак ва =айси адресдан бошланган былса, шу адресдан бошлаб бышатиш керак.

ДЕЛФИ ПРОГРАММАСИНИНГ ТУЗИЛИШИ

Делфидаги ихтиёрий программа лойиҳа файлидан (.dpr кенгайтмага эга) ва бир ёки бир нечта модуллардан (.pas кенгайтмага эга) ташкил топади. Бундай файлларнинг ҳар бири Object Pascalнинг программа бирлиги ҳисобланади.

Лойищанинг тузилиши

Лойиҳа файли Object Pascal тилида ёзилган ва компилятор томонидан қайта ишлаш учун мўлжалланган программа ҳисобланади. Бу программа Делфи томонидан автоматик равишда ҳосил қилинади ва бир нечта қаторни эгаллайди. Унинг программа коди қуйидагича:

```
Program project1;
Uses
    Forms,
    Unit1 in ‘Unit1.pas’ {fmExample}
{$R *.RES}

begin
    Application.Initialize;
    Application.Createform(TfmExample, fmExample);
    Application.Run;
end.
```

Код ойнасида қуюқ ҳарфлар билан хизматчи сўзлар, курсив билан изоҳлар келтирилади.

Модулнинг тузилиши

Модуллар бу программа бирликлари бғлиб, улар программа бўлакларини жойлаштириш учун хизмат қилади. Модулнинг асосий тузилиши қуйидагича:

Unit Unit1;

interface
// Интерфейс эълонлари бўлими

implementation
//реализация бўлими

end.

Интерфейс эълонлари бўлимида программа элементлари тавсифланади, яъни тоифалар, класслар, процедуралар ва функциялар. Уларга бошқа программа модулларидан мурожаат қилиш мумкин.

ПРОГРАММАДАГИ ИЗОЩЛАР

Программани ёзиш давомида дастурловчи изощлардан фойдаланиши ма=садга мувофи=дир. Object Pascalда щам бу имкониятдан фойдаланиш мумкин. Изощларни тавсифлашнинг икки хил кыриниши мавжуд былиб,

1) // белгиси билан бошланган ёзувнинг барчасини изощ сифатида =абул =илади. Масалан,

```
interface
    // Интерфейс эълонлари бўлими
implementation
    //реализация бғлими
```

2) { ва } белгилари орасида келтирилган матнни изощ сифатида =абул =илади. Масалан,

```
Uses
    Forms,
    Unit1 in ‘Unit1.pas’ {fmExample}
```

*{$R *.RES}*

ТИЛНИНГ ОПЕРАТОРЛАРИ

1) Таркибий операторнинг умумий кыриниши =уйидагича:

```
Begin
    . . .
End;
```

2) Шартли операторнинг умумий кыриниши =уйидагича:

if <шарт> then <оператор1> else <оператор1>;

3) Такрорлаш операторлари 3 га былинади:

- Шартсиз цикл операторининг умумий кыриниши =уйидагича:

  for <цикл_параметр>:=<бош_=ий> to <якун_=ий> do <оператор>;

- Олд шартли цикл операторининг умумий кыриниши =уйидагича:

  While <шарт> do <оператор> end;

- Сынг шартли цикл операторининг умумий кыриниши =уйидагича:

  repeat <цикл_танаси> until <оператор>;

4) Киритиш ва чи=ариш операторларининг умумий кыриниши =уйидагича:

Read(); Readln();
Write(); Writeln();

5) танлаш операторининг умумий кыриниши =уйидагича:

case <танлаш_калити> **of** <танлаш_рыйхати> [**else** <операторлар>]
end;

6) ытиш операторининг умумий кыриниши =уйидагича:

goto <белги>;

Белгиларни **label** хизматчи сызи билан тавсифлаб ытилади: **label** <белги1>, <белги2>;

ПРОЦЕДУРА ВА ФУНКЦИЯЛАР

Процедура ва функциялар программага нисбатан алощида ишловчи =исми былиб, у ыз номига эга былади. Программа матнида бу номи келиши процедура ёки функциянинг ча=ирилиши ёки унга мурожаат дейилади. Процедуранинг функциядан фар=и шундаки, функция танасида келтирилган операторлар бажарилгандан сынг щар доим =андайдир =иймат натижа сифатида олинади. Процедура ва функцияларни умуман олганда =исм программа дейилади.

+исм программаларнинг номланиши

Программада ишлатилаётган щар бир исм албатта тавсифланиши шарт былгани каби процедура ва функцияларни щам номини тафсифлаш керак. Тафсифлаш деганда унинг сарлавщаси ва танасини кырсатиш тушунилади. Сарлавщада процедуранинг номи ва унда ишлатиладиган параметри келтирилади. Функцияда эса худди процедурадагидек яна унга =ышимча =илиб функция =айтараётган натижавий =ийматнинг тоифаси щам келтирилади. Сарлавщасидан сынг =исм программанинг тавсифланиш былими ва танаси келади. Тавсифлаш былимида яна =уйиро= даражадаги =исм программалар келтирилиши мумкин. Масалан,

```
Procedure a1;
    Procedurea2;
        . . .
        begin
            . . .
```

```
        end;{a2}
Procedure a3;
        . . .
        begin
                . . .
        end; {a3}
. . .
begin
. . .
end;{a1}
```

+ИСМ ПРОГРАММАЛАРНИ ТАФСИФЛАШ

+исм программа сарлавща ва тана =исмидан иборат. Процедура сарлавщасининг умумий кыриниши =уйидагича:

Procedure <ном> [(<формал_парам_рыйхати>)];

Функция сарлавщасининг кыриниши =уйидагича:

Function <ном> [(<формал_парам_рыйхати >)] : <тоифаси>;

бу ерда <ном> - =исм программанинг номи; <формал_парам_рыйхати > - формал параметрлар рыйхати; <тоифаси> - функция =айтараётган =иймат тоифаси.

+исм программанинг сарлавщасидан сынг =уйидаги директивалардан бири келтирилиши мумкин: Assembler, Forward, Far, Near, External, Inline, Interrupt.

Бу директивалар компилятор щаракатларини ани=лайди ва фа=ат шу =исм программанинг ызига таъсир =илади.

Assembler – =исм программанинг тана =исми ассемблер командалари ёрдамида ёзилишини билдиради.

External - бу директива ёрдамида таш=и =исм программа эълон =илинади.

Far - компилятор узо= ча=иришли моделга мылжалланган =исм программанинг кодини щосил =илади.

Near - компилятор я=ин ча=иришли моделга мылжалланган =исм программанинг кодини щосил =илади.

Forward - =исм программанинг тавсифланиши асосий программанинг пастро\ида келтирилганлигини билдиради.

Inline - =исм программанинг танаси ырнатилган машина инструкциялари (кырсатмалари) ёрдамида ташкил =илинишини билдиради.

Interrupt - узилишларни =айта ишловчи процедура щосил =илишда ишлатилади.

Бу директивалардан таш=ари яна Windowsнинг ядроси билан ишловчи =уйидаги стандарт директивалардан фойдаланиш мумкин: register, pascal, cdecl, stdcall, safecall.

Бу директивалар параметрларни узатишда хизмат =илувчи стек ва регистерларни ишлатади.

+исм программаларда ишлатиладиган параметрлар

Расмий параметрлар рыйхати келтирилиши шарт эмас ва тушириб =олдирилиши мумкин. Агар улар мавжуд былса, унда формал параметрларнинг номлари ва уларнинг тоифалари келтирилади. Масалан,

```
procedure sb (a: real; b:integer; c:char);
```

ёки

```
Function F (a:real;b: real): real;
Function F (a,b: real):real;
```

Бу параметрлар шу =исмпрограмма ичидаги ихтиёрий ифодаларда келтирилиши мумкин.

Расмий параметрлар сифатида =иймат-параметр, ызгарувчи-параметр ёки ызгармас-параметр ишлатилиши мумкин. Ю=оридаги мисолда A ва B =иймат-параметр былиб келган. ЫЗгарувчи-параметр былиши учун параметр олдидан VAR-хизматчи сызини, ызгармас-параметр былиши учун параметр олдидан const-хизматчи сызини =ыйиш етарли. Масалан,

procedure myproc (var A : real; B : real; const c : char);

Бу ерда A – ызгарувчи-параметр, B - =иймат-параметр; C – ызгармас-параметр.

+иймат-параметр ишлатилганда унинг =иймати фа=ат процедурани ичида ызгаради.

ЫЗгарувчи-параметрлар ишлатилса, асосий программадаги =иймати ызгаради.

Object Pascalнинг яна бир имконияти бу нотоифавий параметрлардан фойдаланишдир. Нотоифавий параметр деб шундай параметрга айтиладики, унинг тоифаси процедура сарлавщасида келтирилмайди. Нотоифавий параметр былиб фа=ат ызгарувчи праметр ишлатилади. Масалан,

Procedure myproc (var aParametr);

Тоифаси йы= былган щолларда ишлатилади. Масалан, хотиранинг =айсидир майдонини бош=асига кычириш учун Blockread, Blockwrite, Movememory ва бош=а процедуралар ёрдамида.

Жимлик быйича параметрлар

Умумий кыриниши =уйидагича:

<ном> : <тоифа> = <=иймат>

Масалан,

Procedure p(a: array of integer; s: string=””);

Бу щолда процедурага мурожаат =илинганда

P([1,2,3], ‘’);

ёки

P ([1.2.3]);

деса щам былади. Масалан,

Procedure p (a:array of integer; s: string = ””; b: integer=0);

Бу процедурага =уйидагича мурожаат =илиш мумкин:

P([1,2,3]);
P([1,2,3],’=атор’);
P([1,2,3],’’,1);

Массив-параметрлар ва =атор-параметрлар

Procedure s(a: array [1..10] of real);

Агар массивнинг =андайдир элементи =исм программага юборилиши лозим былса, щеч =андай муаммо ту\илмайди. Лекин агар массивнинг ызи юборилиши керак былса, у щолда даставвал унинг тоифасини ёзиш мумкин. Масалан,

```
Type
    Atype=array [1..10] of real;
    Procedure s (var a: atype);
    . . . . .
```

+аторларда щам шунга ыхшаш:

```
type
        inputtype=string[15];
        outputtype=string[30];
        function st (s : inputtype) : outputtype;
        . . . . .
```

Очи= массивлар

Очи= массивларда унинг ылчами ва чегараларини ани=ламасдан келтирилади. Масалан,

```
Procedure myproc (openArray: array of integer);
```

Очи= массивлар 1 ылчамли былади.

КЛАССЛАР

Класс бу маълумотлар тоифаси, объект эса компьютер хотирасида бор былган класснинг экземпляридир. Классни эълон =илиш учун class хизматчи сызидан фойдаланилади.

Классларнинг майдони, хоссалари ва методлари (класснинг майдон ва хоссларини =айта ишловчи) мавжуд.

Класс щосил =илинганда у хотирада битта экземплярда былади. Класс сифатида тавсифланган ызгарувчилар эса шу класснинг нусхалари щисобланади.

ОБЪЕКТЛИ ДАСТУРЛАШНИНГ УЧ ПРИНЦИПИ

1. Мерос олиш

Щар доим щам класс нусхаларини олиб ишлаш ты\ри былмайди, чунки янги хоссалар =ышилиши ёки айрим мавжуд былганлари керак эмас былиши мумкин. Бу щолда янгитдан класс щосил =илиш ты\ри эмас. Орти=ча керакмас ишлардан =утулиш учун объектли

дастурлашда хосса ва методлардан мерос олиш принципи киритилган. Дастурловчи база классини тавсифлаши етарли. Масалан, «автомобил» база классини. Унга кыра «енгил автомобил» ва «юк автомобили» классини щосил =илиш мумкин. Бунда база классининг барча майдонлари, хоссалари ва методлари мерос олинади. Уларни =ышимча тавсифлаш керак эмас. База классни эса оталик класси щам дейилади.

Мерос олиш кетма-кетлиги чегараланмаган узунликка эга былиши мумкин. Масалан, «юк машинаси» классида яна меросхыр-класслар былиши мумкин, масалан, «Камаз», «Маз» ва бош=алар. Улар ыз навбатида =ышимча хоссалари ва методларига эга былиши мумкин.

2. Полиморфизм

Масалан, «Камаз» классига тааллу=ли ызгарувчига мурожаат былганда «щаракатланиш» методи ча=ирилганда, программа =айси методни ча=иришни, яъни «автомобил» класси методидан, «юк машинаси» ёки «Камаз» класси методидан ча=иришни щал =илиш керак.

Полиморфизм принципига мос щолда бу методни ча=ирувчи ызгарувчининг тоифасига кыра ечим =абул =илинади, яъни агар ызгарувчи «Камаз» типи сифатида тавсифланган былса, айнан шу «Камаз»га мос «щаракатланиш» методи ча=ирилади.

3. Инкапсуляция

Айрим щолларда класснинг бир нечта майдон ва методларини программанинг ихтиёрий =исмида ишлатиш учун, бош=а майдон ва хоссларини эса фа=ат жорий модулда ва класснинг хусусий методларида ишлатиш учун имкон беришга ты\ри келади. Бу шу класснинг асосий хоссаларига эътиборни жалб =илиш имконини беради. Бу ишни инкапсуляция дейилади.

Ижролар

Бу учта фундаментал имкониятлардан таш=ари Delphi мущитида яна Windows мущитидан ёки программанинг ызидан олинадиган

хабарларни =айта ишлаш имконияти мавжуд. Уларни ижролар дейилади.

Классни тавсифлаш

Янги класснинг умумий кыриниши =уйидагича:

```
Type <класс_номи>=class [(оталик_класси_номи)]
        <класс_аъзолари_рый>
end;
```

Агар янги класс оталик классининг барча характеристикаларини мерос =илиб олса, оталик классининг номи келтирилади. Бу щолда янги класс шу класснинг барча оталари характеристикаларини =абул =илади.

Класснинг майдонлари

+уйидаги кыринишдаги мисолни кырайлик:

```
Type
    TmyClass=Class
        AStrField : String;
        AIntField : Integer;
        AObjField : Tobject;
    End;
Var
    Aobject : TmyClass;
Begin
     . . .
    AObject. aIntField:=0;
    AObject.AStrField:='символлар =атори';
    . . . . .
End;
```

Бу мисолдаги AStrField, AIntField, AObjField лар эълон =илинган TmyClass классининг майдонлари щисобланади.

Класснинг методлари

Класснинг методларини =уйидаги мисолдан кырамиз:

```
Type
   TmyClass=Class
        Function MyFunc (aPar : Integer) : Integer;
        Prosedure MyProc;
End;
```

Бу мисолдаги MyFunc, MyProc лар эълон =илинган TmyClass классининг методлари щисобланади. Методларга мурожаат =илиш =уйидагича былади:

```
Var
        AObject : TmyClass;
Begin
        . . .
        Aobject.MyProc;
        . . .
End;
```

Класснинг хоссалари

Класснинг хоссалари майдонларга мурожаатни ташкил этувчи классларнинг махсус механизми хисобланади. Улар асосан property, read, write хизматчи сызлари ёрдамида эълон =илинади. (Property хизматчи сызини ёзиш шарт). Масалан,

```
Type
    TmyClass=class
        IntField : integer;
        Function GetField : integer; Prosedure SetField(value : integer);
        Property IntegerValue : integer read GetField write SetField;
     End;
```

РЕКУРСИЯ

Рекурсия бу щисоблаш жараёнининг шундай ташкиллаштириш усулики, бунда =исмпрограмма ыз ичидаги амалларни =айта-=айта такрорлаш учун ыз-ызига мурожаат =илишидир. Мисол учун факториални щисоблаш программасини кырамиз:

```
Prosedure TfmExample.bbRunClick(sender : TObject);
Function Factorial(N : Word) : Extended;
Begin
        If  N=0 then Result:=1
                else Result:=N*Factorial(N-1)
End;
Var
        N : integer;
Begin
     Try
        N:=StrToInt (Trim(edInput.Text));
      Except
        Exit;
      End;
      lbOutput.Caption:=FloatToStr(Factorial(N))
end;
```

Программа edInput компонентидан N сонини =абул =илади ва lbOutput компонентига N! =ийматини чи=аради. У Factorial(N) функцияси ёрдамида щисобланади. Бу ерда TfmExample - формани номи; bbRunClick - BitBtn компонентини формага =ыйилганидан сынг унга былган ижро.

Программанинг бошида =уйидаги ёзувлар ёзилади:

```
Type
    TFmExample=Class(TForm)
        Panel1 :  Tpanel;
        BbRun : TbitBtn;
        EdInput : Tedit;
        LbOutput : Tlabel;
   Private
        {Private declarations}
```

```
  public
      {public declarations}
  end;
Var
      FmExample : TfmExample;
Implementation
      . . .
End.
```

КЛАССНИНГ МАХСУС МЕТОДЛАРИ

Ихтиёрий класс таркибига иккита махсус метод - конструктор ва деструктор киритилади. TObject классида бу методлар Create ва Destroy деб номланади.

Конструктор объектни динамик хотирада та=симлайди ва шу хотиранинг адресини self номли ызгарувчига ёзиб =ыяди.

Деструктор объектни тыда (рус тилида - куча)дан ычириб ташлайди. Умуман олганда конструктор ва деструктор процедура щисобланади, лекин Constructor ва Destructor хизматчи сызлар билан эълон =илинади. Масалан,

```
Type
    TmeningClassim=Class
        Ab : integer;
        Constructor Create (Value : integer);
        Destructor Destroy;
    End;
```

Объект хосил =илингандан сынг конструкторни ча=ириш ёрдамида объектнинг методлари, майдонлари ча=ирилиши мумкин. Масалан,

```
Var
    MyObject : TmeningClassim;
Begin
    MyObject.ab:=0;{Хато! Объект конструктор ёрдамида щосил
                    =илинмаган}
```

```
    MyObject:= TmeningClassim.Create; {Ты\ри! аввал объект щосил
                        =илиниб, унинг майдонига мурожаат =иламиз}
    MyObject.ab:=0;
    . . .
    MyObject.free; {кераксиз объектни йы= =иламиз}
End;
```

Бу ерда Free аввал объект бор-йы=лигини текширади. Сынгра Destroy деструкторига мурожаат =илинади. Агар объект конструктор ёрдамида щосил =илинмаган былса, Free методи ча=ирилганда хато беради.

Насл-класс конструкторида аввал ызининг отаси конструкторини ча=ириб, сынгра =ышимча амаллар бажариш лозим. Оталик классининг ихтиёрий методини ча=ириш inherited (мерос олган деган маънога эга) хизматчи сызи ор=али амалга оширилади. Масалан,

```
    Constructor TmeningClassim.Create(Value:integer);
    Begin
       Inherited Create; {мерос олган конструкторни ча=ирамиз}
      Ab:=value;{=ышимча амалларни бажарамиз}
    End;
```

НОМИ БИР ХИЛ МЕТОДЛАР

Delphi 4,5,6 да битта класс ичида бир нечта бир хил номли методларни щосил =илиш имконияти пайдо былди. Ю=оридаги мисолда кырсатилганидек, бир хил номли методлар ишлатилганда, насл ызининг оталик методини ты\ридан-ты\ри «кырмайди», фа=ат inherited хизматчи сызи ёрдамида мурожаат =илиши мумкин. Delphi 4 да overload (=айта юклаш деган маънога эга) хизматчи сызи киритилган былиб, уни ёрдамида оталик ва наслдаги бир хил номли методлар кыринадиган былади.

+уйидаги мисолда бир хил номли методларни битта классда ишлатишни кырамиз:

```
    Procedure Tform1.Button1Click (Sender:TObject);
    Begin
```

```
    Close('Символлар =атори')
End;
Procedure Tform1.Button2Click (Sender:TObject);
Begin
    Close (123)
End;
Procedure Tform1.Button3Click (Sender:TObject);
Begin
    Close (20,300)
End;
Procedure Tform1.Button4Click (Sender:TObject);
Begin
    Close
End;
```

Энди Tform1 классининг Private былимига close методининг =уйидаги учта ёзувини =ышиш керак:

```
Private
    {Private declarations}
    Procedure Close (s:string); reintroduce; overload;
    Procedure Close (i:integer); reintroduce; overload;
    Procedure Close (i,j:integer); reintroduce; overload;
```

Ва нищоят Implementation былимида =уйидагилар ёзилиши мумкин:

```
Procedure Tform1.Close (s:string);
Begin
  Caption:=s;
End;
Procedure Tform1.Close (i:integer);
Begin
  Caption:=inttostr(i);
End;
Procedure Tform1.Close (i,j:integer);
Begin
  Caption:=inttostr(i*j);
End;
```

Программани ишлатгач, учта кнопка Tform1 классининг Close методини чаiиради ва ойнанинг сарлавҳасини ғзгартиради, Button4 эса Tform оталик классининг Close методига мурожаат iилади ва ойнани ёпади.

КЛАССНИ ТАВСИФЛАШ

Ихтиёрий щосил =илинадиган класс Published, Private, Protected, Public ва Automated хизматчи сызлари билан ани=ланадиган былимлардан иборат былиши мумкин. Щар бир былимнинг ичида дастлаб майдонлар, сынгра методлар ва хоссалар ани=ланади.

Public былимида келтирилган майдонлар, хоссалар ва методлар программанинг ихтиёрий бош=а модулидан ча=ириб ишлатиш мумкин.

Published былимида нафа=ат бажарилиш этапида балки програмани ташкиллаштириш этапида щам, яъни объектлар инспектори ойнасида щам ишлатилиши шарт былган хоссалар келтирилади. Бу былим фа=ат ностандарт компонентларни ишлаб чи=ишда ишлатилади.

Private былимида келтирилган элементлар фа=ат шу класснинг методлари ичида ва тавсифланган модулдаги =исм программалар ичида мурожаатга эга былиши мумкин.

Protected былими фа=ат класснинг методларида щамда унинг ихтиёрий наслига тааллу=ли.

Automated былими OLE-автоматлаштириш объектлари интерфейсига =ышилиши мумкин былган хосса ва методларни эълон =илиш учун ишлатилади.

Object Pascalда ихтиёрий былимни бир неча маротаба келтириш мумкин, уларнинг келиш тартиби ащамиятли эмас. Ихтиёрий былим быш былиши мумкин. Масалан:

```
Unit Unit1;
Interface
Uses controls, forms;
Type
     Tform1=class(Tform)
          Button1:Tbutton; // бу бғлимга делфи хизмат iилади.
```

```
                    // ундаги элементлар щаммага тааллу=ли.
Private             // бу былим Unit1 модулига тааллу=ли.
        FIntfield: integer;
        Prosedure  setvalue(Value:integer);
        Function  getvalue: integer;
Published           // бу былим ихтиёрий былимга тааллу=ли.
        Property intfield: read GetValue write SetValue;
Protected           // бу былим насл-классларга тааллу=ли.
        Prosedure proc1;
Public              // бу былим ихтиёрий былимга тааллу=ли.
        Prosedure proc2;
End;
Var
        Form1: tform1;
Implementation
Prosedure tform1.proc1;
        Button1.color:=clbtnface;
        Fintfield:=0;
        Intfield:=0;
        Proc1;
        proc2;
end;
begin
        form1.button1.color:=clbtnface;
        form1.Fintfield:=0;
        form1.Intfield:=0;
        form1.proc1; // мумкин эмас!
        form1.proc2; // мумкин.
end.
```

Бош=а модулдан FIntfield га мурожаат мумкин эмас. Бош=а модулдан насл-класси элементларини эълон =илиш учун =уйидагича ёзиш мумкин:

```
Type
    Tform2=class (tform1)
    . . .
Public
```

```
    Prosedure proc1;
    . . .
End;
```

Шундан сынг Unit2 модулида Form2.Proc1; деб мурожаат =илиш мумкин. Класслар =исм программларни тавсифлаш былимида келтирилмайди.

ИНТЕРФЕЙСЛАР

Интерфейслар COM (Component Object Model – объектларнинг компонент модели), CORBA (Common Object Reuiest Broker Architecture – талаб =илинган умумий объектларнинг брокер орхитектураси) технологияларида ва улар билан бо\ланган масофавий мурожаатли технологияларда асосий ырин эгаллайди. Уларнинг асосий вазифаси - клиент машинаси доирасида масофавий объектнинг хоссалари, методлари ва ижроларини тавсифлашдир. Интерфейс ёрдамида клиентнинг программаси масофавий объектга худди ызининг хусусий объекти сифатида мурожаат =илади.

Интерфейсни щосил =илиш ва ишлатиш

Интерфейслар тоифаларни тавсифлашнинг хусусий щоли былиб, улар interface хизматчи сызи ёрдамида амалга оширилади. Масалан,

```
Type
    Iedit=interface
            Procedure Copy; stdcall;
            Procedure Cut; stdcall;
            Procedure Paste; stdcall;
            Function Undo: boolean; stdcall;
    End;
```

Интерфейснинг класслардан фар=и шундаки, уларда майдонлар былмайди ва read ва write былимларида фа=ат методларга мурожаат =илиш мумкин. Интерфейсда эълон =илинадиган барча аъзолари ягона Public былимида келтирилади. Методлар abstract, virtual, dinamic ёки

overvide былиши мумкин эмас. Улар конструктор ёки десктрукторга эга былмайди.

Агар ихтиёрий класс интерфейсни =ылласа, у щолда =уйидагича былиши мумкин:

```
TEditor=class(TinterfacedObject,IEdit);
    Procedure copy; stdcall;
    Procedure cut; stdcall;
    Procedure Paste; stdcall;
    Function Undo: Boolean;stdcall;
End;
```

Оддий классдан интерфейсли класснинг фар=и, унда биттадан орти= оталик интерфейси былиши мумкин. Масалан:

```
Type
    ImyInterface=interface
        Procedure delete; stdcall;
    End;
    ImyEditor=class (TInterfacedObject, Iedit, Imyinterface)
        Procedure copy; stdcall;
        Procedure cut; stdcall;
        Procedure paste; stdcall;
        Procedure undo:boolean; stdcall;
        Procedure delete; stdcall;
    End;
```

Интерфейс классни реализация былимида мос интерфейс методларини ёзиш шарт. Агар интерфейс =ыйидаги кырынишда эълон =илинган былса,

```
IPaint= Interface
    Procedure circlePaint(canva:Tcanvas; x,y,r:integer);
    Procedure RectPaint(canva:Tcanvas; x1,y1,x2,y2:integer);
end;
```

уни ишлатувчи интерфейсли класс эса:

```
IPainter= class(TinterfacedObject,Ipaint)
      Procedure circlePaint(canva:Tcanvas; x,y,r:integer);
      Procedure RectPaint(canva:Tcanvas; x1,y1,x2,y2:integer);
end;
```

Бу щолда implementation былимида методлар реализациясини кырсатиш керак:

```
Procedure Tpainter.CirclePaint(canva:Tcanvas; x,y,r:integer);
Begin
       With Canva do Ellipse(x,y,x+2*r,y+2*r);
End;
Procedure Tpainter.RectPaint(canva:Tcanvas; x1,y1,x2,y2:integer);
Begin
       With Canva do Rectangle(x1,y1,x2,y2);
end;
```

Энди Tpainter классининг интерфейсли классини эълон =илиш мумкин. Масалан, унинг ёрдамида квадрат ва доира чизиш мумкин.

ВАРИАНТЛАР

Вариантнинг асосий хоссалари

Вариант бу variant тоифаси былиб, у Delphi 2 дан бошлаб киритилган. Дастурловчи компиляция жараёнида маълумотларнинг =андай тоифага тегишли эканлигини олдиндан билиб былмайдиган щолатлар учун киритилган. Вариант-ызгарувчи хотирадан 2 байт орти=ча жой эгаллайди. У ерда бу ызгарувчининг ща=и=ий тоифаси ща=ида ахборот жойлашади. Бу ахборот компиляторга программани ишлатиш жараёнида код щосил =илишга имкон беради.

Вариант-ызгарувчиси =уйидагиларни ыз ичига олади:

- бутун ёки ща=и=ий сон;
- манти=ий =иймат;
- =атор;
- ва=т ва/ёки сана;
- OLE-объект;

- ю=орида санаб ытилган тоифалардан бирига тегишли былган ихтиёрий ылчам ва узунликдаги массив.

Масалан, v-вариантгга v:=’1.0’ =иймат берилган былса, у щолда v+1 ифода бизга 2.0 натижани беради, агар v:=’матн’ =иймат берилган былса, у щолда 1+v ифодадан сынг Evariant Error хатоси ща=ида ахборот беради.

Вариант тоифасининг =уйидагича структураси былиши мумкин:

```
TvarData = packed record
        Vtype:Word;
        Reserved1, Reserved2, Reserved3 : Word;
        case integer of
              varSmallInt : (vSmallInt: SmallInt);
              varInteger : (vInteger: Integer);
              varSingle : (vSingle: Single);
              varDouble : (vDouble: Double);
              varCurrency : (vCurrency: Currency);
              varDate : (vDate: Double);
              var Olestr : (vOlestr: PwideChar);
              var Dispatch : (vDispatch: Pointer);
              var Error  : (vError: WordBool);
              var String : (vString: Pointer);
            var Array : (vArray : PvarArray);
            var ByRef : (vPointer : Pointer);
end;
```

Булардан ташқари яна қуйидаги тоифалардан бирин қабул қилиши мумкин: varEmpty, varNull, varBoolean, varUnknow, varByte.

Вариант тавсифланганда, дастлаб varEmpty да щеч =андай =иймат былмайди. Программа ишлаши жараёнида компилятор вариант =ийматини беради.

Вариантларни бош=а турдаги маълумотлар тоифасига ыгириш

Вариантларни бош=а ызгарувчиларнинг тоифалари =ийматларига ыгирганда =уйидаги =оидаларга эътибор бериш керак:

Нимага олиб келади	Вариантдаги маълумотлар тоифаси					
	VarEmpty	Бутун	Ща=и=ий	Сана- ва=т	+атор	Манти=ий
Бутунга	0	Мос тоифага ыгиради	я=ин бутунгача яхлитлайди	я=ин бутунгача яхлитлайди	бутун тоифага ыгиради	False учун 0, акс щолда -1
Ха=и=ийга	0,0	Мос тоифага ыгиради	Мос тоифага ыгиради	Мос тоифага ыгиради	Ща=и=ий тоифага ыгиради	False учун 0, акс щолда 1
Сана-ва=тга	30.12.1899 00:00:00	Double га ыгиради	Double га ыгиради	-	Санага ыгиради	Double га ыгиради
+аторга	Быш =атор	Символли кыринишда ыгиради	символли кыринишда ыгиради	символли кыринишда ыгиради	-	False учун 0, акс щолда -1
Манти=ийга	False	0 учун False, акс щолда True	0 учун False, акс щолда True	0 учун False, акс щолда True	‘False’ва 0 учун False, акс щолда True	-

Бу ерда бутун сифатида - VarByte, VarSmallInt, VarInteger, VarError;
Ща=и=ий сифатида - VarSingle, VarDouble, VarCurrency;
+атор сифатида - VarString, VarOlestr олинган.

Вариантлар билан ишловчи =исм программалар

+уйидаги =исм программалардан фойдаланиш мумкин:

Function VarAsType (const v: variant; varType: Integer): variant;

v-вариантдаги ахборотни varType параметри билан ани=ланадиган ахборотга ыгиради.

Procedure varCast (var Dest: variant; const source: variant; varType: integer);
Бу процедура Sourceдагини VarTурега ыгириб, Dest ызгарувчисига ёзиб =ыяди.

Procedure VarClear (var v: variant);

Динамик хотирани агар у вариант билан бо\ланган былса, бышатади ва унга varEmpty тоифасини беради.

Procedure varCopy (var Dest: varian; const source: variant);

Sourse вариантни Dest вариантга кычириб =ыяди.

Function VarFromDateTime (DateTime: TdateTime): variant;

DateTimeдаги ахборотни вариант сифатида =айтаради.

Function VarIsEmpty(const v:variant):Boolean;

Агар V да ахборот былмаса, True натижани олади.

Function VarToStr(const v:variant): string;

V ни =аторга ыгиради.

Function VarToDateTime(const v: variant): TdateTime;

V ни сана-ва=т тоифасига ыгириб беради.

Function VarType(const v: variant): integer;

V даги ахборотнинг тоифасини =айтаради.

Вариант массивлар

Вариантнинг =иймати сифатида ахборот массивлари келиши мумкин. Уларни вариант массивлар деб аталади. Вариант массив элементларининг =иймати сифатида VarString дан таш=ари ихтиёрий вариант тоифасини олиши мумкин. Вариант массивнинг элементлари сифатида вариантлар щам келиши мумкин, яъни бундай массивнинг элеменлари турли тоифадаги ахборотга эга былиши мумкин. Масалан,

var

```
   v : variant;
begin
    // бир ылчамли вариант массив щосил =иламиз:
    v:=varArrayCreate ([0,4], varvariant);
    // Уни тылдирамиз:
    v[0]:=1;                    // бутун тоифали
    v[1]:=1234.5678;  // ща=и=ий тоифали
    v[2]:='Hello world';  // =атор
    v[3]:=True;          // манти=ий
    // 5-элементи былиб яна массив щосил =иламиз:
    v[4]:=VarArrayof([1,10,100,1000]); // hello world
    Caption:=v[2];
     lbOutput.Caption:=IntToStr(v[4][2]); // 100
 end;
```

ФАЙЛЛАР ТОИФАЛАРИ

Файл тоифасини =уйидаги уч хил усул билан щосил =илиш мумкин:

<ном> = file of <тоифаси>;
<ном> = TextFile;
<ном> = file;

Бу ерда <ном> - файл тоифасининг номи,
File of – хизматчи сызлар,
TextFile - матнавий файллар тоифасининг стандарт номи,
<тоифаси> - Object Pascalнинг ихтиёрий тоифаси.
Масалан,

```
type
    Product= record
        Name: String;
        Code: Word;
        Cost: Comp
     end;
     Text 80 = file of string[80];
var
   F1 : file of Char;
```

```
F2 : TextFile;
F3 : File;
F4 : Text 80;
F5 : file of  Product;
```

Тавсифлаш усулига кыра 3 хил кыринишдаги файлларни ажратиш мумкин:

1. Тоифавий файллар (file of сызи билан берилади).
2. Матнавий файллар (Textfile тоифаси билан берилади).
3. Нотоифавий файллар (File тоифаси билан берилади).

Ю=оридаги мисолда F1, F4 ва F5 лар - тоифавий файллар, F2 - матнавий файл, F3 - нотоифавий файл.

ФАЙЛЛАРГА МУРОЖААТ

Программада файлга мурожаат =илиш учун файлни очиш процедураси хизмат =илади. У янги файл очиш, ёки борини очиш, файлдан ы=иш ёки ёзиш учун ишлатилади.

assignFile (<ф.ы.>, <файл номи>);

Бу ерда <ф.ы.>-файл ызгарувчиси, у программада файл тоифаси ызгарувчиси сифатида эълон =илинади. <файл номи> - матнавий ифода былиб, унда файл йыли кырсатилади.

Файлни инициаллаш деганда, унга маълумотларни узатиш йыналишини кырсатиш тушунилади.
Файлни Object Pascalда файлни ы=иш учун, ахборот ёзиш учун ёки бир ва=тнинг ызида щам ы=иш, щам ёзиш учун очиш мумкин.

Файлни ы=иш учун стандарт процедура ёрдамида инициалланади:

Reset (<ф.ы>);

Бу ерда <ф.ы.>- файл ызгарувчиси былиб, assignFile да эълон =илинган файл билан бо\ланади.

Бу процедура ишлатилганда кырсаткич файлнинг бошини кырсатиб туради. Агар йы= файлга мурожаат =илинаётган былса, хато ща=ида ахборот чи=аради. Файл бор ёки йы=лигини билиш учун FileExist стандарт функцияси ишлатилади. У файл бор былса True, акс щолда False =ийматни =айтаради. Масалан,

```
begin
      if  FileExist(FileName) then
     . . . . . // файл бор былса
     else
      . . . . . // файл былмаса
end;
```

Object Pascalда Reset процедураси билан очилган тоифавий файлларга Write процедураси билан ахборотни ёзиш мумкин. Reset билан очилган матнавий файллар учун Write ёки Writeln процедурасини ишлатиш мумкин эмас.

Rewrite (<ф.ы>);

стандарт процедураси <ф.ы> билан бо\ланиб, унга ахборот ёзишни инициация =илади. Агар бор файлга ёзилаётган былса, уни йы= =илиб устига ёзиб юборади.

Append (<ф.ы>);

стандарт процедураси мавжуд файлга ёзишни инициация =илади. Бунда файл кырсаткичи унинг охирига ырнатилади. Бу процедура фа=ат матнавий файл учун ыринли, яъни программада унинг тоифаси TextFile былиши керак.

Агар матнавий файл Reset ёки Rewrite билан очилган былса, Append процедурасини ишлатиш шу файлнинг ёпилишига ва =айтадан очилишига олиб келади. Бунда ёзувларни =ышиш учун очилади.

Файл очилганда унга шу файлнинг тизимли дескриптори, яъни Handle билан бо\ланади. Уни FileSetTime ва FileGetTime функцияларига мурожаат =илишда ишлатилади.

FindNext, FindClose, FindFirst =исм программалари файллар гурущига мурожаат =илиш учун имкон беради. Улар умумий белгилар билан бирлаштирилган былиши керак, яъни файлнинг таниш маскаси ва атрибутлари кырсатилади.

Маска сифатида =уйидаги белгилардан фойдаланиш мумкин: *,?

Масалан, *. * - ихтиёрий файлни =идириш;
*.pas – pas кенгайтмали ихтиёрий файл;
a*.* - боши а щарфли ихтиёрий файл;
a?.dat – боши а, ундан кейин бир щарфли dat кенгайтмали файл.

Агар файлнинг йыли кырсатилмаса, жорий каталогдан =идиради.

FindFirstга мурожаат былганда, Attr параметри иккилик разрядини са=лайди, улар =айси файлга мурожаат рухсат =илинганини билдиради.

SysUtils модулида файл атрибутлари =уйидагича эълон =илинади:

```
const
    faReadOnly=$01;  // фа=ат ы=иш учун
    faHidden=$02;    // беркитилган файл
    faSysFile=$04;   // тизимли файл
    faVolumeID=$08;  // том идентификатори
    faDirectory=$10; // =исм каталог номи
    faarchieve=$20;  // архив файл
    faAnyfile=$3F;   // ихтиёрий файл.
```

Комбинациялардан фойдаланиш мумкин. Масалан, $06 – барча беркитилган ва/ёки тизимли файлларни танлайди.

FindFirst процедураси сифатида TsearchRec тоифасидаги ызгарувчи =айтарилади. У =уйидагича ани=ланади:

```
type
```

```
TsearchRec=record
        Time: integer;
        Size: integer;
        Attr: integer;
        Name: TfileName;
        ExcludeAttr: integer;
        FindHandle: Thandle;
        FindDate: Twin32FindDate;
end;
```

Бу ерда Attr- файл атрибути,
Time-файл щосил =илинган ёки янгиланган ва=т ва сана,
Size-файлнинг байтлардаги узунлиги,
Name-файлнинг номи ва кенгайтмаси,
FindDate-файл ща=ида =ышимча ахборотни са=лайди.

Мисол:

```
procedure TfmExample.bbRunClick(Sender:Tobject);
var
        mask: string;
begin
        mask:=edInput.Text;
        if Mask=’’ then
        Mask:=’*.*’;
        mmOutput.Lines.Clear;
        if FindFirst(Mask,faanyFile,Sr)=0 then
        repeat
                mmOutput.Lines.Add(Sr.Name);
        until FindNext(SR)<>0
        FindClose(SR);
end;
```

WINDOWSНИНГ ФАЙЛЛАР БИЛАН ИШЛОВЧИ ВОСИТАЛАРИ

Windows операцион системаси файллар билан ишловчи воситаларга эга, улар Delphi дастурида Windows модулларига ишора =илингач, ишга тушиши мумкин былади.

Асосийлари =уйидагилар:

CopyFile –бир файл ичидагиларни бош=асига кычиради.
CreateDirectory - дискда янги каталог щосил =илади.
CreateFile – янги ёки бор файни очади.
DeleteFile – файлни ычиради.
FindClose - FindFirstFile ва FindNextFile функциялар учун ажратилган хотирани бышатади.
FindNextFile – кейинги файлни =идиради.
GetBinaryType – файлнинг бажарувчи файллилигини ани=лайди. Агар бажарувчи файл бор былса, унинг тоифасини =айтаради.
GetDiskFreeSpace – дискда =анча быш жой борлигини байтларда =айтаради.
GetFileAttributes – файл атрибутларини =айтаради.
GetFileSize – файл ҳажмини =айтаради.
GetFileType – файл тоифасини =айтаради.
LockFile – файлга бош=а программалардан мурожаат былишидан са=лайди.
MoveFile – файл ёки каталогни номини ызгартиради.
OpenFile – мавжуд файлни очади.
ReadFile – файлдан маълумотларни ы=ийди.
WriteFile – файлга маълумотларни ёзади.
UnlockFile - lockfile функцияси билан щимояланган файлнинг щимоясини олиб ташлайди.

Файлни щосил =илиш ва уни очиш

+уйидаги функция ёрдамида щосил =илинади:

Function FileCreate (Const FileName: Stirng):intejer;

Бу функция файл идентификаторини =айтаради (ихтиёрий мусбат сон), агар файл щосил =ила олмаса –1 =айтаради.

Файлнинг очилиши =уйидаги функция билан бажарилади:

Function FileOpen (Const FileName: String; mode: Longword): intejer;

Очиш режимини Mode параметри билан ани=ланади. Масалан:

- FmOpenRead - фа=ат ы=иш учун очиш;
- fmOpenwrite -фа=ат ёзиш учун очиш;
- fmOpenreadwrite – щам ы=иш, щам ёзиш учун очиш;

Маълумотларни файлдан ы=иш учун =уйидаги функциялар ишлатилади:

Function FileRead (Handle: intejer; var Buffer;
Count: intejer): intejer;

Бу ерда Count - параметри ы=илган байтлар сонини кырсатади; Buffer - бу ызгарувчига шу байтлар ёзилади.

Маълумотларни файлга ёзиш учун =уйидаги функциялар ишлатилади:

Function filewtite (Handle: intejer; const Buffer;
Count: intejer): intejer;

Bufferда ёзилиши керак былган маълумотлар са=ланади.
Файлни ёпиш учун =уйидаги процедура ишлатилади:

Procedure FileClose (Handle: intejer);

+уйидаги мисолда янги файл щосил =илиниб, унга матнавий =атор ёзилади:

```
Var
    FileHandle: intejer;
    S: String;
Begin
    S:=’матнавий =атор‘;
    FileHandle:=fileCreate ('C:\TMP\S.txt');
    FileWrite (FileHandle, S, Sizeof (S));
```

```
        FileClose (FileHandle);
    End;
```

Файлни =идириш учун =уйидаги функция ишлатилади:

```
    Fuction fileSearch (const name, DirList: string): String;
```

Бу ерда Name - =идирилаётган файл номи; DirList - каталоглар рыйхати.

Бу функция файлнинг тыли= йылини =айтаради ёки агар файл топилмаса быш =аторни =айтаради. Масалан:

```
    S:= FileSeach (‘myDoc.doc’,'C:\мои документы; D:\info \texts’);

    If  s<> ‘‘ then ShowMessage(S)
                else ShowMessage (‘бундай файл топилмади ‘);
```

Файлнинг ихтиёрий позициясига ытиш учун =уйидаги фукция ишлатилади:

```
    Function FileSeek (handle, offset, origin: intejer ): intejer;
```

Originга нисбатан offsetда кырсатилганча байтга сурилади. Origin =уйидаги =ийматга эга былиши мумкин:
0 - файл бошидан
1 - жорий позициядан
2 - файл охиридан.

АССЕМБЛЕРНИ ЧА=ИРИШ

Ассемблерга мурожаат =илишнинг умумий кыриниши =уйидагича:

```
    Asm
        командалар
    End;
```

Масалан, x ызгарувчисига y ызгарувчисини =ышишни =уйидаги усуллар ёрдамида билан амалга ошириш мумкин:

A)
```
var x, y, z : intejer;
. . .
asm
        mov eax, x
        add eax, y
        mov z, eax
end;
```

b)
```
asm
        mov z, x+y
end;
```

Иккита ызгарувчини бир-бирига кыпайтириш учун эса =уйида келтирилган иккита функция ор=али бажариш мумкин:

a)
```
function mult (x,y : intejer): longint;
    begin
        result := x*y;
    end;
```

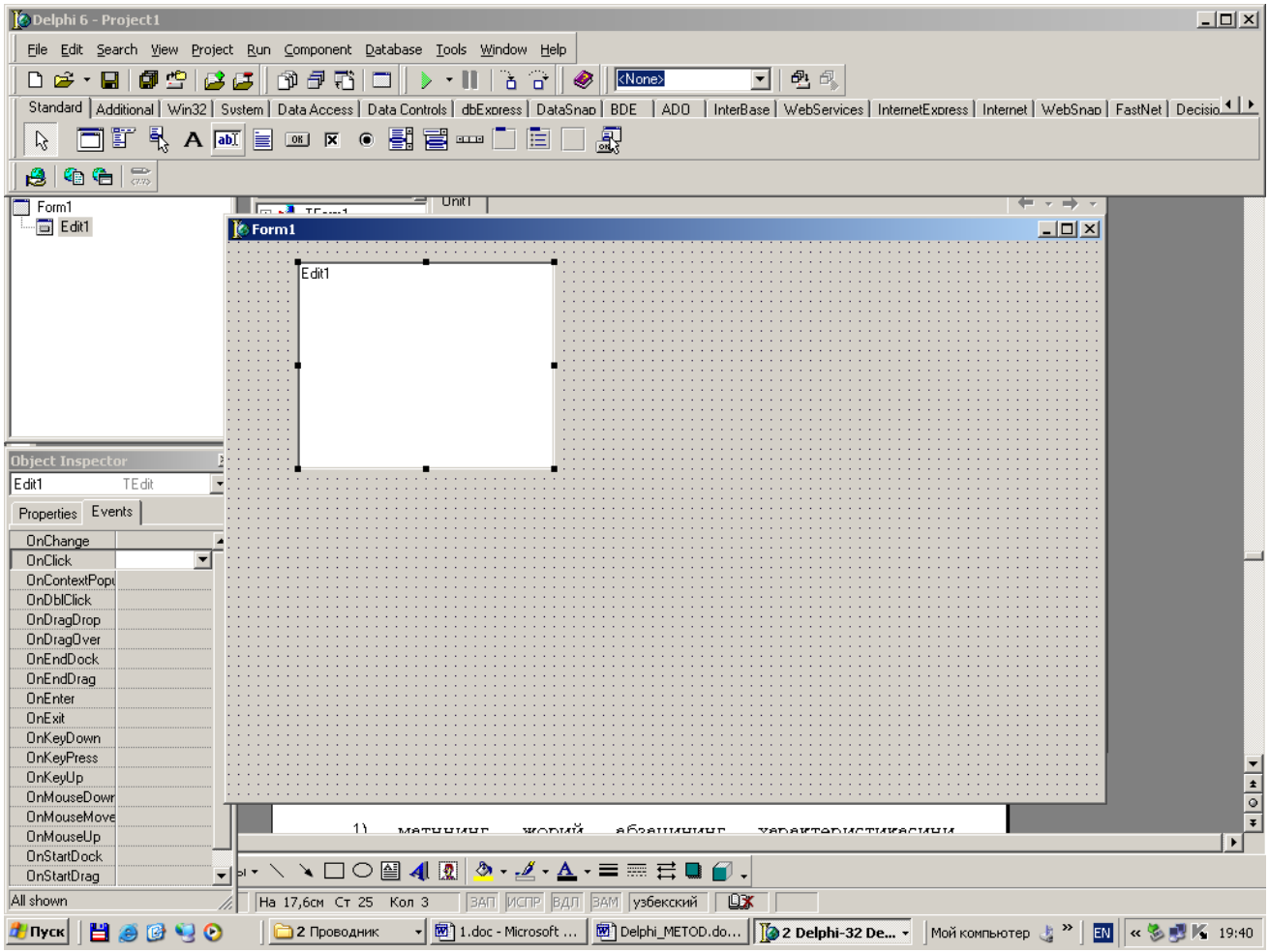
b)
```
function mult (x,y : intejer): longint;
    asm
        mov eax, x
        imul y
    end;
```

Биринчи келтирилган функция ёрдамида ызгарувчиларни кыпайтириш учун 13 та команда ишлайди, чунки Delphiдаги командалар ассемблер тилига ыгирилади. Иккинчи функция эса бевосита ассемблер командалари ор=али ёзилгани учун кыпайтириш амали 2 та команда билан бажарилади.

МАТН МУҲАРРИРИ КОМПОНЕНТИ

Матн муҳаррири компоненти TRichEdit компонентидир. У Standart саҳифасида жойлашган. TMemo компонентига нисбатан бу компонентнинг бир мунча афзалликлари мавжуд. Масалан, у .rtf форматидаги файлларни =ыллаб-=увватлайди щамда матннинг абзацлари устида форматлаш ишларини амалга оширади.

Матн мущаррири компонентини щосил =илиш учун формага TrichEdit компонент ырнатилади (15-расм).



15-расм. Матн мущаррири компонентини щосил =илиш учун формага TrichEdit компонентини ырнатиш.

Матн муҳаррири компонентини ишлатиш

Бу компонентнинг асосий хусусияти бу алощида абзацларни форматлаш имкониятининг мавжудлигидир. Бундай форматлаш компонентнинг иккита хусусияти быйича бажарилади:

1) матннинг жорий абзацининг характеристикасини ани=ловчи - paragraph.
2) белгиланган матн характиристикаларини ани=ловчи SelAttributes.

Paragraph хоссаси TparaAttributes типига эга. Унинг асосий хоссалари =уйидагилар:

Alignment - текислаш. =иймати сифатида taLeftJustify (чап томонга), taCenter (ыртага), taRightJustify (чап томонга);
Tab - сурилишлар массиви. Tab клавиша босилганда курсорнинг силжиши;
TabCount - tab массиваги элементлар сони.

SelAttributes хоссаси TtextAttributes тоифасига эга былиб, ыз навбатида Tfont классини тавсифловчи хоссалардан иборат: Charset (символлар тыплами), Color (ранг), Height (баландлиги), Name (номи), Pitch (фиксирланган ёки ызгарувчан кенглик), Size (ылчам), Style (стил) дан иборат.

+ышимча хоссаси Protected Boolean =ийматга эга былиб, белгиланган матнни тащрирлашга йыл бермайди.

Курсор щолатини ы=иш учун GetCaretPos методи хизмат =илади. Ажратилган матнни нусхасини ызгарувчига кычириш учун GetSelText методи хизмат =илади. +уйидаги мисолни кырамиз:

```
Pocedure Tform1.Button1Click (sender:Tobject);
      Begin
           With RichEdit1 do
        begin
                Paragraph.Numbering:=nsBullet;
                Paragraph.Alignment:=taCenter;
             End;
      End;
Procedure Tform1. Button2Click (sender:TObject);
      Begin
```

```
        With RichEdit.SelAttributes do
        Begin
              color:=clred;
              Style:=[fsBold];
              Size:=14;
              Name :=’Courier’;
        End;
  End;
```

TRichEdit компонентининг ызига хос хусусияти бу .rtf форматидаги матнни юклаш ва са=лашидир. Бунинг учун Plaintext хоссаси =ийматини =ийматини False деб =ыйиш керак. масалан,

```
  Procedure Tform1. Button3Click (sender:TObject);
   Begin
        RichEdit1.Plaintext:=false;
        RichEdit1.lines.SaveToFile (‘test.rtf’);
    End;
```

МОДУЛЛАР

Модулларнинг тузилиши =уйидагича:

```
Unit <ном>;
Interface
      <интерфейс =исми>
Implementation
      <бажарилувчи =исм>
Initialization
      <инициализация =илиш =исми>
Finalization <якунловчи =исм>
End;
```

Модулларнинг сарлавщаси ва уларнинг ызаро бо\ланиши

Модулнинг сарлавщаси Unit хизматчи сызидан кейин келади. Унинг номи диск файлининг номи билан бир хил былиши керак. Масалан:

Unit Global;

Берилган былса, шу модулнинг матни Global.Pas диск файлида са=ланиши керак. Модул номи модуллар билан ызаро бо\ланиш учун ва асосий дастур билан бо\ланиш учун хизмат =илади. Бу бо\ланиш махсус сыз билан бо\ланади. Масалан:

Uses <модуллар_рыйхати>

Мисол учун, Uses Windows, System, SysUtils, Global;

Модуллар ыз ичида бош=а модулларни ишлатиши мумкин. Uses хизматчи сызи interface хизматчи сызидан кейин ёки Implementation хизматчи сызидан кейин келиши керак. Ёки иккаласидан кейин келиши мумкин.

Interface =исми

Бу =исмида модулнинг барча глобал объектларини (тоифалар, ызгармаслар, ызгарувчилар =исм дастурлар) тавсифлаш керак былади. У асосий дастурдан ёки бош=а модуллардан мурожаат =илишга мылжалланган. Масалан,

```
Unit Cmplx;
Interface
    Type
        Complex=record
            Re, im: real
        End;
        Function AddC (x,y: Complex ): Complex;
        Function MulC (x,y: Complex ): Complex;
```

Агар бош=а модулда Uses Cmplx; деб ёзилса, унда ыша модулда Complex тоифаси ва AddC ва Mulc функцияларни Cmplx модулидан олиб ишлатиш мумкин былади.

Бажарилувчи =исми

Бажарилувчи =исми Implementation былимидан кейин ёзилади. Унда модулда ишлатиладиган локал объектлар тавсифланиши мумкин, яъни ёрдамчи тоифалар ызгармаслар, ызгарувчилар ва =исм дастурлар щамда белгилар. Масалан,

```
Unit Cmplx;
Interface
  Type
        Complex=record
              Re, im: real;
End;
   Function AddC (x,y: Complex ):Complex;
   Function MulC (x,y: Complex ):Complex;
Implementation
  Function AddC (x,y: Complex ):Complex;
        Begin
              . . .
        End;
  Function MulC;
        Begin
              . . .
        End;
End.
```

Интерфейс =исмида берилган =исм дастурлар албатта Implementation былимида келтирилиши керак.

ДИНАМИК БОҒЛАНАДИГАН КУТУБХОНАЛАР (DLL)

Динамик бо\ланадиган кутубхоналар Dinamic Link Library сызларидан олинган былиб, бош=а программистлар томонидан ёзилган процедура ва функцияларни ёки Object Pascal дан бош=а тилларда ёзилган процедура ва функцияларни ызимизнинг программамизга улаш имконини беради.

DLL да программага ызгарувчилар, ызгармаслар, тоифаларни узата олмайди.

Ёзилган модуллар программа билан компоновка жараёнида бо\ланади, яъни статик бо\ланади. DLL программаларни ызини алощида ишлатиб былмайди).

Реализация (ишлатиш)

Object Pascalда DLL щосил =илиш учун Library хизматчи сызидан фойдаланилади. Ундан сынг унинг номи келади. Умумий кыриниши:

Library <ном>;

Лекин модулдан фар=ли томони шундаки, Library хизматчи сызидан кейин келган ном шу файлнинг номи билан бир хил былиши шарт эмас.

DLL матнининг структураси оддий программаникига ыхшаш былади, лекин бажарилувчи операторлар шу кутубхонани юклаш жараёнида бир марта бажарилади. Щар гал кутубхонага мурожаат былганда ишора щисобчиси бирга оширилади ва операторлар бажарилмайди.
DLLни тавсифлаш былимида тоифалар, класслар, ызгармаслар ва ызгарувчилар эълон =илиниши мумкин. Улар ча=ираётган дастур учун беркитилган былади ва фа=ат DLLнинг ичида ишлатилиши мумкин. Бу былимда яна экспорт =илинаётган =исмдастурларни эълон =илиш махсус былими щам келтирилади. Бу былим exports хизматчи сызи билан бошланиб, =исм дастурда "," ор=али санаб ытилади. Масалан,

```
Library myLibrary;
      Function myFunc(. . .): . . . ;
       Begin
            . . .
         End;
      Procedure myProc;
         Begin
             . . .
         End;
      exports
            myFunc, myProc;
```

```
Begin
    . . .
End.
```

Dll да бир нечта exports рыйхати келтирилиши мумкин, лекин уларда эълон =илинаётган =исм дастурлар ю=ориро=да келтирилиши лозим.

+исмдастурни номидан таш=ари яна DLL сарлавщасида унинг тартиб номери, яъни бутун сонли индекси келтирилиши лозим. Бу шу =исм дастурни ча=ираётган пайтда унинг номига эмас, унинг индексига ишора =илади ва ва=тдан тежайди. У =уйидагича тартибланади:

```
Exports
    MyFunc index 1, MyProc index 2;
```

Бунда индекс учун 0÷32767 гача былган бутун сонларни ишлатиш мумкин.
Дастурловчи экспорт =илинаётган =исмдастурнинг таш=и номини ызгартириш щу=у=ига эга. Бу эса Name хизматчи сызи ор=али амалга оширилади. Таш=и ном албатта “ ‘ “ белгиси ичига олиб ёзилиши шарт. Масалан,

```
Exports
    MyProc index 2 name ‘Newproc’;
```

Таш=и номлар катта ва кичик щарфлари фар=ланади. Ча=ираётган дастур export =илинаётган =исмдастурнинг номига ёки индексига ишора =илади. Экспорт қилинаётган қисм дастурларни эълон қилишнинг турли усулларини кўрсатувчи DLL ни ҳосил қилишга мисол кўрамиз. Мисол сифатида Cmplx номли модули танланган бўлиб, унинг таркибига 4 та процедура киритилган бўлиб, улар комплекс сонлар билан амаллар бажаришга мўлжалланган.

Мисол учун, кутубхона модулини щосил =илиш учун File | New | Unit командаси берилади ёки репозиторий ойнасида Dll пиктограммасига босилади. Бу ерда шу проектнинг номи келтирилади. Масалан,

```
Library  Cmplx;
Uses
        SysUtils, Classes;
Type
        Tcomplex=record
                Re, im: real;
End;
Function addC (x,y: Tcomplex): Tcomplex; stdcall;
        Begin
                Result.im : =x.im+y.im;
                Result.re : =x.re+y.re;
        End;
Function subc (x,y: Tcomplex): Tcomplex; stdcall;
        Begin
                Result.Im := x.Im-y.Im;
                Result.re := x.re+ y.re;
        End;
Function mulC (x,y: Tcomplex): Tcomplex; stdcall;
        Begin
                 Result.re := x.re*y.re+x.im*y.im;
                 Result.im := x.re* y.re-x.im*y.im;
        End;
Function divC  (x,y: Tcomplex): Tcomplex; stdcall;
        Var
                Z : real;
      Begin
                Z := sqr(y.re)+sqr(y.im);
                Try
                        Result.re:=(x.re*y.re+x.im*y.im)/z;
                        Result.im: =(x.re* y, im-x.im*y.re)/z
                Except
                        Result.re: =1e+309;
                        Result.im: =1e+309;
                End;
        End;
Exports
                Addc index 1 name ‘Addc’ resident,
```

```
        Subc index 2,
        Mulc index 3,
        divc index 4;
begin
end.
```

Эътибор қилинг, бизнинг DLL даги барча функциялар stdcall билан берилган. Бу API Windows32 нинг янги функциялари билан мос келишини таъминлайди.

Агар биз stdcall ни бермасак, компилятор ундан кўра самарилоқ бўлган register ни ишлатади, лекин бу ҳолда Delphi дан бошқа тилларда ёзилган дастурлардан DLL га мурожаат қила олмас эдик.

Маслаҳат: Агар сиз Delphi дан бошқа дастурлардан DLL га мурожаатни таъминламоқчи бўлсангиз, қисмдастурларни stdcall ёки safecall директиваси билан беринг.

Dll дан =исмдастурларни ишлатиш учун external хизматчи сызидан фойдаланилади. Масалан,

```
Procedure MyProc; external ‘myDll’; ёки
Procedure MyProc; external ‘myDll’ index 2; ёки
Procedure MyProc; external ‘myDll’ name ‘Extname’;
```

Шундан сынг экспорт =илинаётган =исмдастур Object Pascal даги оддий =исм дастурга ыхшаб ча=ирилиши мумкин.

Фойдаланиш

Статик юкланиш

+уйидаги дастурда юқорида келтирилган Cmplx кутубхонаси ишлатилади:

```
Type
    Tcomplex=record
        Re, im: real;
    End;
    Function AddC(x,y: Tcomplex): Tcomplex; stdcall; external ’cmplx’;
    Function SubC(x,y: Tcomplex): Tcomplex; stdcall; external ’cmplx’;
```

```
    Function mulC(x,y: Tcomplex): Tcomplex; stdcall; external 'cmplx';
    Function DivC  (x,y: Tcomplex): Tcomplex; stdcall; external 'cmplx';
    Procedure TfmExample.bbRunClick(Sender: TObject);
    Var
        x, y, z: Tcomplex;
    End;
End.
```

EXCEPTION КЛАССИ

Object Pascal да истисноларни қайта ишлаш учун ҳимояланган бўлим механизми киритилган:

1. **try**
   <оператор>
   except
   <истиснони қайта ишловчилар>
   else
   <оператор>
   end;

2. **try**
   <оператор>
   finally
   <оператор>
   end;

except бўлимида истисно қайта ишловчиларининг умумий тузилмаси қуйидагича:

on <истисно класси> **do** <оператор>;

бу ерда **on, do** – хизматчи сўзлар; <истисно класси> - истиснони қайта ишловчи класс; <оператор> - Object Pascal даги ихтиёрий оператор Goto дан бошқа.

Бу класс барча истисно-класслар учун оталик класси щисобланади. Delphiда стандарт иснисно-класслар ани=ланган былиб, уларга =уйидагилар мисол была олади:

Класс	Отаси	Иснисноии =айта ишлаш
EarrayError	Exception	Массивлар билан ишлашдаги хатоликлар юзага келганда
EbrokerException	Exception	Браузер объект-серверни топа олмаганда
Eprinter	Exception	Windows принтерда хато былганида хабар беради
EcommonCalendarError	Exception	Ноты\ри сана киритилганда TcommonCalendar класси объектларида ва унинг наслларида
EconvertError	Exception	StrToInt, StrToFloat функцияларини =айта ишлаганда хато
EcorbaException	Exception	Corba технологиясини ишлатганда программада хато чи=са
EdataBaseError	Exception	Маълумотлар базасида компонент хато топганда
EdateTimeError	Exception	Ноты\ри ва=т ёки сана киритилганда TdateTimePicker компоненти объектида
EdivByZero	EIntError	Нолга былишда хато
EFCreateError	EStreamError	Файлни щосил =илишда хато. Масалан, йы= былган

		каталогда очиш ёки фа=ат ы=иш учун былган файлни очиш
EoleCtrlError	Exception	Программа ActiveX элементи билан бо\лана олмаганда

Масалан,

```
Try
      Reset(F);
      While not EOF(F) do
      Begin
            . . .
      end;
      CloseFile(F)
      Except
      On E: EinOutError do
                  Showmessage (‘файл билан ишлашда’+
            IntToStr(E.ErrorCode) +’номерли хато содир былади’);
End;
```

Истиснони ча=ириш

Айрим щолатларда программист хусусий истиснони щосил =илишига ты\ри келади. Бунинг учун у Raise хизматчи сызини ишлатиши мумкин. Агар бу сыз try билан exception орасида келса ёки try билан finally орасида келса, у щолда ыша защоти except ёки finally былими ишга тушади.

Агар Raise сызи except билан end орасида келса ёки finally билан end орасида келтирилган былса, бу бўлимлар ишини тугатган щисобланади ва кейинги операторга бош=ариш узатилади. Масалан,

```
Raise EinOutError.Create ('хато!');
```

Хусусий истисно классини щосил =илишга мисол

Масалан,

```
Type
      EIntCheckError =  class (Eabort)
End;
Var
      F:TextFile;
      S:String;
      K:integer;
Begin
      . . .
      Try
      AssignFile (F,FileName);
      Reset (F);
      While not EOF (F) do
      Begin
      Readln (F,S);
        K:=StrToInt (S);
        If k<o then
            raise EintCheckError.Create('манфий сон');
       if k<=o then
            raise EintCheckError.Create('мусбат былмаган сон');
      . . .
      end
      except
      on E:EIntCheckError do ShowMessage (E.Message);
      on EInOutError do ShowMessage('ноты\ри файл амали');
      on EConvertError do ShowMessage('сонни ёзишда хато');
      on EintError do ShowMessage('=айта ишлашда хато');
      end;
end;
```

Бу мисолда хусусий истисно класси щосил =илиниб, уни ёрдамида =андайдир файл очилиб, бу файлга =андайдир соннинг ёзилиш амалининг бориши ща=ида ахборот берилмо=да.

ГРАФИК ИНСТРУМЕНТАРИЙ

Windowsнинг тасвирий бойликлари график =урилма DC (Device Context, яъни контекст дескриптори) билан ва унга кирувчи 3 та инструмент - шрифт, перо ва мый=алам билан бо\ли=дир.

Delphiga Windowsнинг график инструментларини ишлатишни осонлаштирувчи махсус класслар мавжуд. Булар Tcanvas, Tfont, Tpen, Tbrush.

Tcanvas - контекс учун;
Tfont-шрифт учун;
TPen-перо учун;
Tbrush - мый=алам учун.

Бу класслар билан бо\ли= объектлар Canvas, Font, Pen, Brush хоссалар ор=али программада ишлатилиши мумкин.

Tfont класси

Бу класс ор=али ихтиёрий график =урилма (экран, принтер, плоттер ва бош=алар) учун шрифт-объект щосил =илинади. TFont классининг хоссалари:

Charset - символлар тыплами;
Color - шрифт ранги;
FontAdapter - шрифт ща=идаги ахборотни ActiveX компонентига =ыяди;
Handle –шрифтнинг дескриптори;
Height - шрифтнинг экран пикселларидаги баландлиги;
Name – шрифт номи;
Pitch - щарфларнинг мантда жойлашиш усулини ани=лайди.
Style - шрифтнинг стили. Унинг =ийматлари fsBold, fsItalic, FsUnderline, fsStrikeOut (устидан чизилган).

TPen класси

Бу класс ёрдамида чизи=ларни чизиш учун перо объекти щосил =илинади.Хоссалари:

Color - чизи=ларнинг ранги;
Handle - перо дескриптори;
Mode - чизи=ларнинг фон билан бо\ланишини ани=лаш усули;
Style - чизи=ларнинг стилини ани=лайди;
Width – чизи=нинг кенглиги.

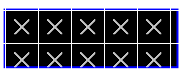
Чизи=ларнинг турлари =уйидагича былиши мумкин:

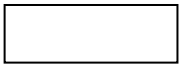
_____ psSolid
- - - - - psDach
---------- PsDot
--------- PsDashDot
---------- PsDashDotDot
psClear
_____ psInsideFrame

TBrush класси

Бу класс ёпи= фигураларнинг ичини быяш учун хизмат =илади. Бу класснинг хоссалари =уйидагилар:

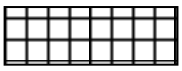
Bitmap - растрли тасвирни са=лайди. Агар бу хосса ани=ланган былса, color ва style хоссалари ишлатилмайди.
Color - мый=алам ранги
Handle - дескриптори
Style - мый=аламнинг стили. Уларнинг =иймати =уйидагича былиши мумкин:

 - BsSolid

 - bsclear

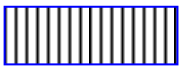
 - bsBDiagonal

 - BsFDiagonal

 - bsCross

 - bsDiagCross

 - BsHorisontal

 - bsVertical

TCanvas класси

Бу класс "канва" ни щосил =илади. У перо, мый=алам ва шрифт билан чизиш имконини беради. Font, Pen, Brush объектларини инкапсуляция =илади.

Хоссалари:

Brush - мый=алам-объект
ClipRect - быялиши лозим былган майдоннинг жорий ылчамларини ани=лайди
CopyMode - растрли тасвирни фон ранги билан билан ызаро бо\ли=лигини ырнатади
Font - шрифт-объект
Handle - канва дескриптори
Pen - перо-объект
Pixels - канта массиви пикселлари

Методлари:

Arc (x1,y1,x2, y2, x3, y3, x4, y4); - (x1,y1) ва (x2,y2) ты\ри тыртбурчакни ыз ичига олувчи эллипс ёйини чизади.
Draw (x, y, Graphic); – график объектни чизади.
Ellipse (x1, y1, x2, y2); – (x1,y1) ва (x2,y2) ни ыз ичига олувчи эллипс чизади.
LineTo (x,y); - чизи= чизади.
MoveTo (x,y); - перо щолатини ызгартиради.
Rectangle (x1,y1,X2,y2); - ты\ри тыртбурчак чизади ва быяйди.

TGraphics класси

Бу класс абстракт класс щисобланади ва ыз-ызидан ишлатилмайди. Унинг асосида программада график объектлардан фойдаланиш учун бош=а класслар щосил =илинган. Класснинг методлари абстракт ва виртуал былиши мумкин.

Унинг хосалари:

Width, Height - объектнинг кенглиги ва баландлиги (пикселларда).
Modified - агар объект ызгартирилган былса, true =иймат олади.
Palette - Windows ранг палитрасининг идентификатори
Transparent - агар объект кыринмас (яъни прозначний) режимда чизилса true =иймат олади.

Абстракт методлари:

LoadFromFile - график ахборотни файлга юклайди.
SaveToFile - график ахборотни са=лайди.
LoadFromClipboard - график ахборотни windowsнинг алмашиниш буферига юклаш.
SaveToClipboard - график ахборотни windowsнинг алмашиниш буферида са=лаш.

TGraphics классининг насллари
TBitmap класси (ну=тали тасвир)

TBitmap класси махсус класс былиб, уни ёрдамида график ну=тали тасвирларни (яъни .bmp кенгайтмали) Windows алмашиниш буферидан ёки файлдан юклаш, са=лаш мумкин. Шунингдек улар устида =ышимча амаллар бажариш мумкин.

Хоссалари:

Canvas - тасвир майдони былиб, унда чизиш мумкин.
Empty -агар объектда щеч =андай тасвир былмаса, true =ийматни олади.

PixelFormat - пикселдаги битлар сони (рангнинг чу=урлиги).
ScanLine - ну=тали тасвирнинг щар бир =аторига массив кырсаткич. Массивдаги элементлар сони height нинг =ийматига тенг. Кырсаткич ёрдамида ани= бир пикселга мурожаат =илиш мумкин. Масалан,

```
Var BitMap: TbitMap; P: PbyteArray;
    . . .
    P:=BitMap.ScanLine[y];
    For x:=0 to BitMap.Width-1 do
    P[x]:=color;
```

TransparentColor - тасвирни экранга чи=аришда кыринмас ранг.
TransparentMode - кыринмас рангни ани=лаш усули.

Методлари:

FreeImage - ну=тали тасвирни са=лаш учун ранг чу=урлигини камайтириш йыли билан хотира майдонини камайтириш.
LoadFromResourseID - тасвирни программа ресурсларидан юклаш.
Mask - тасвирдаги ани= рангни кыринмас ранг сифатида ырнатиш.
Бу класс ёрдамида унча =ийин былмаган спрайтли мултипликацияни щосил =илиш мумкин. Бунинг учун дастлаб компьютер хотирасига олдиндан тайёрлаб =ыйилган фонли тасвир ва бир неча спрайт расмлар керак былади. Сынг Tbitmap классининг динамик щосил =илинган объектидан фойдаланилади. Унга дастлаб фон кычирилади, сынг унга алощида спрайтлар =ыйилади. Маълумотлар чизиш мумкин былган объект майдонга =ыйилади (canvas хоссаси). Бунда draw, copyrect ва бош=а методларни ишлатиш мумкин. Шундан сынг щосил былган расм экранга чи=ади. (Timage компонентининг picture хоссасининг bitmap хоссаси) ва экранда кырилади.

TIcon класси

Бу класс Windowsнинг значок форматли (.Ico кенгайтмали) тасвирлар билан ишлаш учун =ылланилади. Унинг хосса ва методлари bitmap классидан фар= =илмайди, фа=ат значок щар доим ани= кыринмас рангга эга былади, ва уни Stretchdraw методи билан масштаблаштириш мумкин эмас.

TmetaFile класси

Windowsда график маълумотларнинг махсус тоифааси - метафайл (.emf ва .wmf кенгайтмали) мавжуд. У ну=тали тасвирдан шу билан фар= =иладики, пикселларни ырнида махсус код са=лаган былади. Экранга чи=арилаётганда, масалан, А ну=тадан В ну=тага =изил чизи= билан ёки ты\ри тыртбурчакни чизсин каби ишлайди.

Махсус методлари:

Description - ички изощ.
Enhanced - метафайл .EMF форматида са=ланган былса, true =ийматни олади.
Inch - метафайл координата системасига кыра дюмдаги ну=талар сони.
Mmwidth, mmheight - тасвирнинг шартли ну=талардаги (0,01 мм) баландлиги ва кенглиги.

TJPEGImage - JPG форматидаги тасвир класси

Бу класс JPEG тасвирлари билан ишлаш учун =ылланилади. У катта ра=амларни ихчам кыринишда са=лаш учун хизмат =илади. Бу класс билан ишлаганда canvas хоссасини ишлатиш мумкин эмас. Бу класс тасвирни экранда кырсатиш учун =ылланилади.

Айрим хоссалари:

CompressionQuality - тасвир са=ланган файл сифати ва ылчами орасидаги нисбат. Унинг =иймати бирдан минггача былиши мумкин. Бундан катта =иймат файлнинг кам жой олишига, лекин сифати бузилишига олиб келади.
Crayscale - агар тасвир кулранг шкалада (255 та ранг) экранга чи=арилаётган былса, true =ийматни олади. Бу эса расмни очишни тезлаштиради.
Performance - Тасвирни файлдан ы=иганда жуда яхши сифати- jpBestQuality ёки жуда ю=ори тезлиги- jpBestSpeed.
Pixelformat - jpeg расм форматга кырсатгич (8 битли ёки 24 битли).

Progressivedisplay - агар экранга чи=арилаётган тасвир экранда =исм-=исм былиб кырсатилса, true =иймат олади.

Progressivencoding - расмни =исм-=исм =илиб чи=аришга рухсат былса, true =иймат олади.

Scale - тасвир масштаби =ийматлари - jsfullsize (тыли= ылчами), jsHalf (ярим ылчамда), jsQuarter (чорак ылчамда).

Smoothing - агар тасвир =исм-=исм былиб чи=арилганда, унинг тини=лиги ошадиган былса, true =ийматни олади, акс щолда тасвир кичик былаклар билан =аторма-=атор чи=ади.

Методлари:

Compress - compressionQuality хоссаси =ийматига кыра тасвирни си=иш.

DIBNeeded - jpeg тасвирини .bmp форматига айлантириш.

JpegNeeded - икки битли формат .bmp асосида jpeg тасвирини щосил =илиш.

TPaintBox - расм чизиш майдони компоненти

У System панелида жойлашган былиб, кыпинча формада бир нечта расм чизиш майдонларини ажратиш учун хизмат =илади. Унда ягона canvas хоссаси бор. Битта щодисаси OnPaint ни ишлатади.

Сич=ончанинг кырсатгичлари

Сич=онча кырсатгичи экран быйича силжиганда унинг шакли ыз щолатини ызгартириши мумкин. Delphiда бу ишни компонентнинг cursor хоссасига =араб бажарилади, яъни

```
property Cursor:Tcursor;
Type Tcursor=-32768÷+32767;
```

Барчамизга маълумки, сич=онча кырсатгичининг бир нечта стандарт кыринишлари мавжуд. Дастурлаштириш амалиётида кыпинча сич=онча кырсатгичи щолатини программанинг щар хил ойналари учун ызгартириш эщтиёжи ту\илади. Масалан, =андайдир узо=ро= жараён бажарилиши давомида сич=онча кырсатгичи crHaurGlass

кыринишини олади, жараён тыхтагач эса, кырсатгич яна ызининг дастлабки щолатига =айтади. Дастурдаги барча ойналар учун кырсатгични шаклини бир ва=тнинг ызида ызгартиришда глобал объект Screen нинг Cursor хоссасидан фойдаланилади. Масалан,

```
Screen.Cursor:=haurglass; //Узо=ро= жараён
Screen.Cursor:=Default;  //Дастлабки щолатга ытади
```

Дастурловчи ностандарт кырсатгични тайёрлаши ва уни ишлатиши мумкин. Буниг учун у биринчидан Delphiнинг тасвир мущарири ёрдамида кырсатгич тасвирини чизиши ва бу тасвирни ва дастурнинг ресурс файлига жойлаштириши лозим. Иккинчидан, ишловчи дастурда дастлаб ресурс файлидан кырсатгични loadcursor функцияси ёрдамида юклаш керак ва уни Screen объектининг Cursor рыйхатида регистрация =илиш лозим былади. Шундан сынг эса компонентнинг ёки экраннинг cursor хоссаларини ностандарт кырсатгич =илиб =ыйиш мумкин.

Ностандарт кырсатгични щосил =илиш жараёнини =уйида кырамиз.

1. Асосий менюдан Tools/ Image Editor командаси берилганда Delphiнинг тасвир мущарририни юклайди.
2. Мущаррир ойнасида File/ New/ Resourse File, сынгра Resourse/New/ Cursor танланади.
3. Resourse/ Rename команда бериб ва албатта Cursor1 ырнига янги ном бериш керак.
4. Ресурслар рыйхатидаги Resourse га икки марта сич=онча билан босилади. Ёки Resourse/ Edit команда билан бажариш мумкин. Бунда экранда 2та быш майдонли мущаррир ойнаси пайдо былади. Бу икки ойна =изил рамкага олинган былади. Бунда чап ойнада чизиш учун мылжалланган былади. Ынг ойнада эса оригинал кырсатиб турилади, яъни табиий кыринишда.
5. Чап ойнада ызингиз щощлаган ихтиёрий тасвирни чизиш мумкин былади.
6. Untitled1.res га бориб сич=онча кырсатгичи босилади. Сынг са=лаб =ыйиш учун File/ Save ёрдамида ресурс файлини Cursor номи билан са=лаймиз.

7. Янги лойища щосил =илиш керак ва унга Form1 формаси учун onCreate щодисаси =айта ишлагични =уйидаги кырнишда ёзиш керак былади.

```
{$R *.DFM}
{$R Cursor. res}
Procedure Tform1.FormCreate (Sender: Tobject);
 Begin
      // кырсатгични регистрация =иламиз:
      Screen.Cursors[1]:=loadCursor (Hinstance, ‘myCursor’);
      // уни форманинг клиент =исми учун ишлатамиз
      Cursor:=1;
End;
```

Дастурдаги {$R Cursor.res} ёзувни унутиш керак эмас, чунки компоновкаловчи Cursor.res файлини дастурнинг ресурс файлига бо\лаб =ыйиши учун –22 дан –1 гача былган диапазондаги =ийматларни стандарт кырсатгичларга ажратилганлиги сабабли, бу сонларни дастурда бериб былмайди.

DELPHIDA МАЪЛУМОТЛАР БАЗАСИ (МБ)

Delphiни RAD (Rapid Application Development) системаларга киритишади. Унда замонавий МБ ни бош=ариш тизимлари (МББТ) нинг барча имкониятларидан фойдаланиш мумкин. МБ га сыровларни визуал тайёрлаш, SQL тилида ты\ридан-ты\ри сыровлар ёзиш мумкин.

Delphi ёрдамида локал щамда масофавий МБ билан ишловчи иловалар, шунингдек, интернетда МБ ни нашр этиш щам мумкин.

МБ га кириш

Умумий тушунчалар

Ахборот тизимларининг замонавий шакли бу маълумотлар банклари былиб, у =уйидагиларни ыз ичига олади:

1) щисоблаш тизимлари;
2) Маълумотлар базасини бош=ариш тизимлари (МББТ);
3) Бир ёки бир нечта МБ;

4) Амалий программалар пакети (МБ иловалари).

МБ ахборотни са=лаш учун хизмат =илади, шунингдек, уларга =улай ва тез мурожаатни таъминлайди. МБ турли тоифадаги маълумотларнинг бирлашмасидан ташкил топади. Бу маълумотлар ани= =оидаларга кыра ташкиллаштирилади. МБ даги ахборот

1) манти=ан ты\ри;
2) оши=ча эмас;
3) бир бутун былиши керак.

МББТ МБ ни щосил =илиш, тылдириш ва фойдаланиш учун мылжалланган дастурий ва тил воситаларининг тыпламидан иборат былади. МББТ ишлатилишига кыра шахсий ва оммавийга былинади.

Шахсий МББТ битта компьтерда ишловчи локал МБ ни щосил =илиш имконини таъминлайди. Буларга Paradox, dBase, FoxPro, Access ва бош=алар киради. Access 97 ва Access 2000 да маълумотларга оммавий мурожаатни ташкил этиш мумкин.

Оммавий МББТ “клиент-сервер” архитектурасида ишловчи ахборот тизимларини щосил =илиш имконини беради. Буларга Oracle, Informix, SyBase, MS SQL Server, InterBase ва бош=алар киради.

МБ ни ташкиллаштириш

Маълумотларнинг ташкиллаштирилиш турига кыра МБ нинг асосий =уйидаги моделлари фар=ланади:

1) иерархик;
2) тармо=ли;
3) реляцион;
4) объектга йыналтирилган.

Иерархик моделда маълумотлар дарахт кыринишида тасвирланади. Лекин =ийин манти=ий бо\ли=ликка эга былган маълумотлар билан ишлаганда бу модел жуда катталашиб кетади.

Тармо=ли моделда ихтиёрий граф кырининшида ташкиллаштирилади. Бу моделнинг камчилиги структурасининг =атъийлиги ва уни тузишнинг ю=ори =ийинчилигидир.
Бу иккала моделнинг яна бир камчилиги бу маълумотлар структураси МБ ни лойищалаштириш бос=ичида ани=ланади ва уларга мурожаат былганда ызгартирилмайди.

Объектга йыналтирилган моделда МБ ни алощида ёзувлари объект кырининшида тасвирланади. МБ ёзувлари ва уларни =айта ишлаш функциялари ыртасида объектга йыналтирилган дастурлаш тиллари воситаларига мос келган бо\ланишлар ырнатилади. Бу моделлар тармо=ли ава реляцион моделларнинг хусусиятларини ыз ичида мужассамлаштиради ва =ийин тузилишли маълумотлар ишлатилган йирик МБ ни щосил =илиш учун ишлатилади.

Реляцион моделда МБ жадвалларнинг ызаро муносабатлар ор=али бо\ланишига кыра ташкиллаштилади. Бу моделнинг эътиборга олйи= томони, тузилишининг оддийлиги, мослашувчанлиги, компьютерда ишлатишнинг =улайлиги, назарий ёзилишнинг мавжудлигидир. Кыпгина замонавий МБ реляциондир.

Ахборот тизимларининг архитектураси

МБ ва иловаларни ызаро жойлашишига кыра =уйидагиларга ажратиш мукин:

1) локал МБ;
2) масофавий МБ.

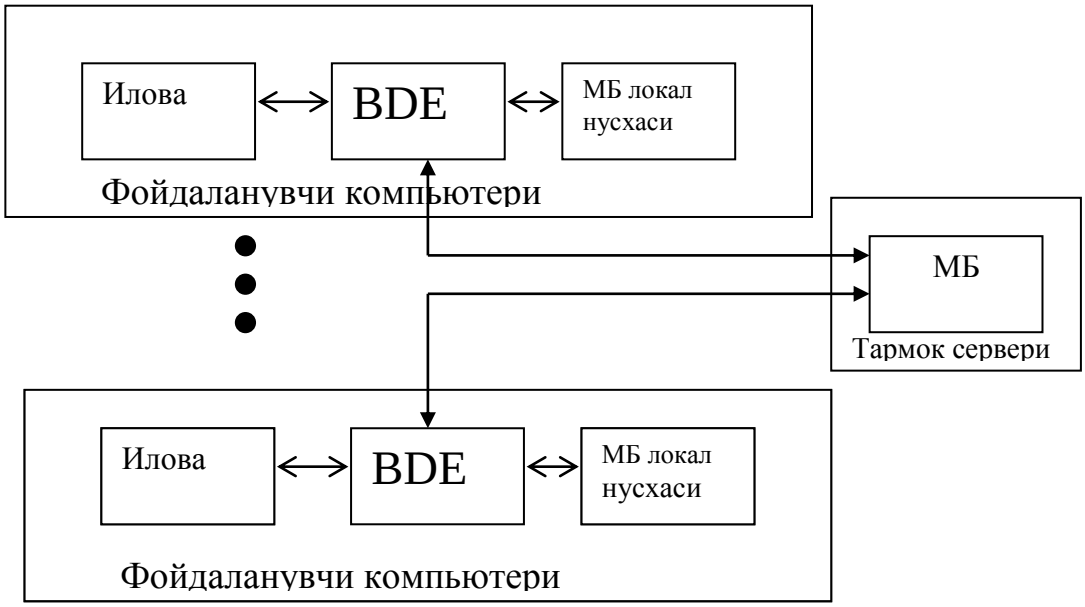
Локал МБ устида амаллар бажариш учун локал иловалар щосил =илинади ва ишлатилади (17-расм). Масофавий МБ устида амаллар бажариш учун клиент-серверли иловалардан фойдаланилади (18-расм).

Delphi-илова МБ га BDE (Borland DataBase Engine- Borland фирмасининг МБ процессори) ор=али мурожаатни амалга оширади.

BDE маълумотларга мурожаатни таъминловчи динамик кутубхоналар, драйверлар тыпламидан иборат (16-расм).

16-расм. Фойдаланувчи компьютери

Локал МБ га мурожаат =илиш учун BDE МБ процессори dBase, Paradox, FoxPro МБ лари форматидаги щамда матнавий файллар билан ишлаш имконини берувчи стандарт драйверлардан фойдаланилади. Локал МБ дан тармо=да фойдаланилганда унга кып фойдаланувчиларнинг мурожаати амалга оширилиши мумкин.



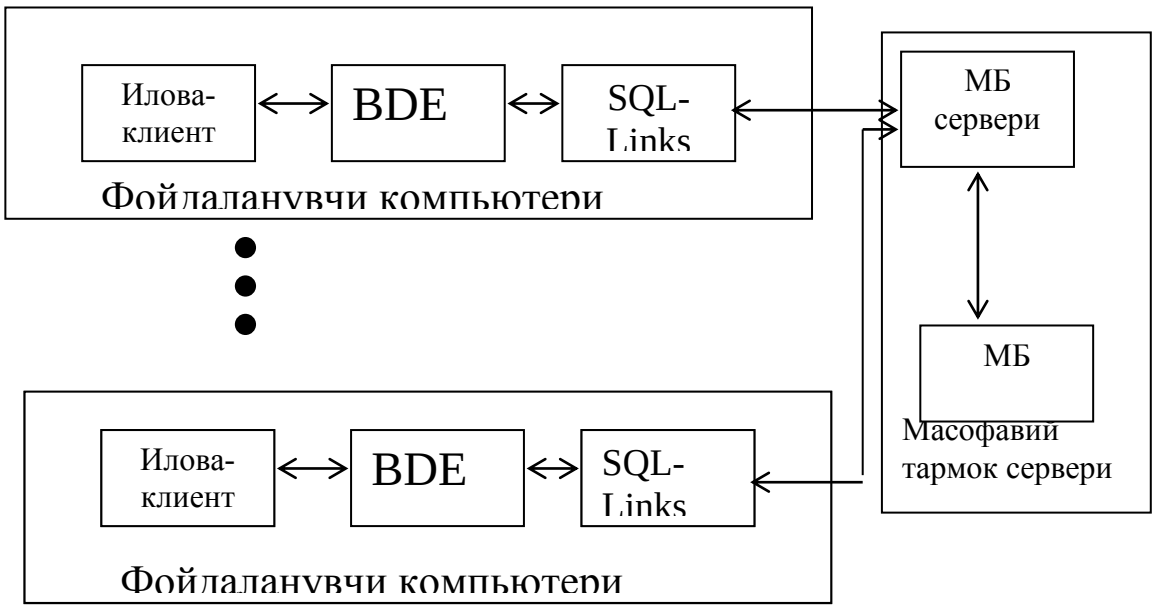
17-расм. «файл-сервер» архитектураси.

МБ файллари ва улар билан ишловчи иловалар бу щолда тармо= серверида жойлашган былади. Бунда щар бир фойдаланувчи ызининг компьютерида серверда жойлашган илованинг нусхасидан фойдаланади. Локал МБдан бундай тармо=ли фойдаланишни «файл-сервер» архитектураси дейилади. «Файл-сервер» архитектурасининг ызига хос камчиликлари мавжуд былиб,

1) кып фойдаланувчилар тармо==а мурожаат =илганда жадвалнинг барчасини щар бир фойдаланувчининг компьютерига кычириш керак былади. Бунда тармо==а былган юкланиш ошиб, ишнинг секинлашишига олиб келади.

2) битта фойдаланувчи ызгартирган ызгартиришлар бир муддат бош=аси учун =орон\и былади. Шунинг учун МБ ни тез-тез янгилашга ты\ри келади. Бундан таш=ари битта фойдалаувчи ишлатаётган жадвал ёзувларини бош=аси учун блокировка =илиш керак былади.
3) МБ ни бош=ариш алощида компьютерлар ор=али былгани сабабли, мурожаатни текширишни ташкил =илиш бир мунча =ийинлашади щамда МБ нинг конфиденциаллигини таъминлаш имконияти камаяди.

Масофавий МБ тармо= сервери компьютерида жойлаштирилиб, шу МБ билан ишлашни ташкил =илувчи илова эса фойдаланувчи компьютерида жойлашса, бундай архитектурани «клиент-сервер» архитектураси дейилади.



18-расм. «Клиент-сервер» архитектураси.

Ахборот тизими сервер ва клиент =исмларга былинган былади. Сервер компьютери клиентдан алощида жойлашган былганлиги учун уни яна масофавий сервер деб щам аталади.

Клиент бу фойдаланувчининг иловасидир. Маълумотлар олиш учун клиент масофавий серверга сыровни щосил =илади ва узатади. Сыров

SQL (Structural Query Language сызларидан олинган) тилида ёзилади. Бу тил серверга мурожаат =илишнинг стандарт воситаси щисобланади.

Масофавий сервер сыровни олгандан сынг уни SQL-серверга йыналтиради (яъни МБ серверига). SQL-сервер махсус программа былиб, у масофавий МБни бош=аради ва сыровни бажарилишини щамда натижаларни клиентга жынатишни таъминлайди.

Шундай =илиб, «клиент-сервер» архитектурасида клиент маълумотларни олиш учун сыров беради ва фа=ат шу сыровга ты\ри келган ахборотни олади. Сыровни =айта ишлаш масофавий серверда амалга оширилади.

Бунда ызига хос хусусиятлари бор былиб, булар:

1) тармо==а былган юкланиш камаяди, чунки бунда фа=ат зарур ахборот билан ишланади;
2) ахборот хавфсизлигининг ошишига олиб келади, чунки серверда жойлашган ягона программа билан бош=арилади.
3) Клиент иловалари =ийинчиликларининг камайиши.

«Клиент-сервер» архитектурасини щосил =илиш учун кыпинча Oracle ёки MS SQL Serverдан фойдаланилади. Бундай МББТни саноат МББТ дейилади.

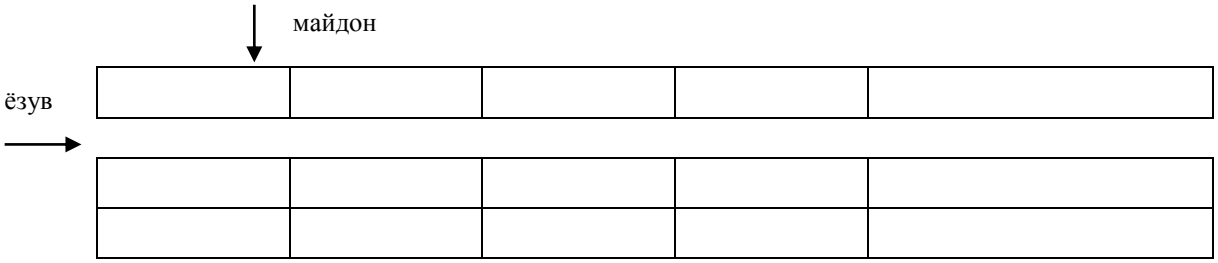
Delphi-иловаларнинг бундай саноат МББТга мурожаати SQL Links драйверлари ор=али амалга оширилади. Delphiнинг ызининг InterBase МББТ мавжуд былиб, унга SQL Links драйверларини ишлатиш керак эмас.

Реляцион МБ

МБ ни щосил =илувчи жадваллар =атти= дискда каталогда жойлашади. Жадваллар файлларда са=ланади ва оддий электрон жадвалга (MS Excelдагига) ыхшайди (19-расм).

Майдон матнавий, бутун сонли ёки ва=т кыринишидаги тоифалардан бирининг =ийматларидан бирининг маълумотларидан ташкил топган былиши мумкин. Жадвалдаги майдонга =иймат киритилганда

автоматик равишда =ийматни ва майдон тоифасининг мослигини текширади.



19-расм. Реляцион МБ жадвали структураси.

Жадвалда камида битта майдон былиши керак. Жадвалнинг тузилиши =уйидагилардан ташкил топади:

1) майдонларнинг тавсифи;
2) калит;
3) индекслар;
4) майдонлар =ийматларига чегараланишлар;
5) жадвалларараро ишора бутунлигига чегараланишлар;
6) пароллар.

Жадваллар устида =уйидаги амалларни бажариш мумкин:

1) щосил =илиш (тузилишини ани=лаш);
2) тузилишини ызгартириш;
3) номини ызгартириш;
4) ычириш.

Маълумотларга мурожаатнинг усуллари

Улар =уйидагиларга былинади:

1) кетма-кет;
2) ты\ридан-ты\ри;
3) индексли кетма-кет.

Жадвалларараро бо\ли=лик

Хусусий щолда МБ фа=атгина битта жадвалдан, масалан, ташкилот ходимларининг ту\илган кунларидан ташкил топган былиши мумкин. Лекин реляцион МБ ызаро бо\ланган жадваллардан ташкил топади. Жадвалларарo бо\ланишни ташкил этишни бо\лаш ёки жадвалларни бирлаштириш дейилади.

Жадвалларарo бо\ли=ликни МБ ни щосил =илиш ва=тида ёки МББТ иловаларининг бажарилиши жараёнида ырнатиш мумкин. Иккита ёки ундан кып жадвалларни бирлаштириш мумкин. Реляцион МБ да бо\ланган жадваллардан таш=ари алощида щеч =айси жадвал билан бо\ланмаган жадваллар щам былиши мумкин.

Жадвалларни бирлаштириш учун бо\ланиш майдонларидан фойдаланилади. Бо\ланиш майдонлари албатта индексланган былиши шарт. Асосий жадвал билан бо\ланиши лозим былган жадвалдан индекс ёки таш=и калит олинади. Бу индексни майдонларининг таркиби бутунлай ёки =исман асосий жадвалнинг индекс майдонлари таркиби билан мос келиши керак.

Жадвалларарo бо\ли=ликнинг =уйидаги турлари мавжуд:
1) «бирга-бир» кыриниш;
2) «бирга кып» кыриниш;
3) «кыпга-бир» кыриниш;
4) «кыпга-кып» кыриниш.

Бо\ланган жадваллар билан ишлаганда =уйидагиларга эътибор бериш керак:
1. Бо\ланиш майдонини ызгартирганда (тащрирланганда) жадвал ёзувлари орасидаги бо\ланиш бузилиши мумкин. Шунинг учун асосий жадвал ёзувининг бо\ланиш майдонини тащрирлагач, мос равишда унга быйсунувчи жадвалларнинг бо\ланиш майдонлари =ийматларини щам ызгартириш керак.
2. Асосий жадвалдаги ёзувни олиб ташлаганда, унга мос келувчи быйсунувчи жадваллардаги ёзувларни ычириб ташлаш керак.
3. Быйсунувчи жадвалга янги ёзув киритилса, унинг бо\ланиш майдони асосий жадвалнинг бо\ланиш майдонининг =ийматига тенг =илиб ырнатиш керак.

МБ БИЛАН ИШЛОВЧИ ВОСИТАЛАР

Delphiда МБ билан ишловчи воситаларни умуман 2 та катта гурущга былиш мумкин:

1. инструментал воситалар;
2. компонентлар.

МБ билан амалларни бажариш учун Delphi тизими =ыйидаги инструментал воситаларни таклиф =илади:

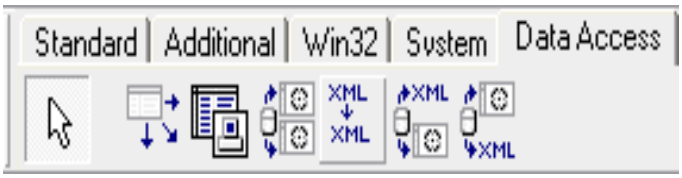
1. Borland DataBase Engine (BDE);
2. BDE Administrator- BDE нинг турли хил параметрларини ырганиш учун утилита;
3. Database Desktop- жадвалларни, SQL-сыровларни ва QBE-сыровларни щосил =илувчи ва тащрирловчи программа;
4. SQL Explorer - МБ проводниги. У МБ ни кыриш ва тащрирлаш имконини беради;
5. SQL Builder - SQL-сыровларни визуал тузишни ташкил =илувчи программа;
6. SQL Monitor - Масофавий МБ га SQL-сыровларнинг бажарилиш тартибини текшириб борувчи программа;
7. Data Pump - МБ орасида ахборотни узатиш учун мылжалланган программа;
8. InteBase Windows Interactive SQL (WISQL)- масофавий МБ ни бош=ариш учун программа.
9. InterBase Server Manager - масофавий сервердан бош=ариш учун программа.
10. SQL Links- масофавий саноат МББТ га мурожаат учун драйверлар. Масалан, Microsoft SQL Server ёки Oracle. Delphi билан текинга тар=атиладиган InterBase саноат серверига SQL Links драйверини ишлатмасдан щам ташкил этиш мумкин.
11. Local InterBase Server - Borland InterBase SQL серверининг локал версияси.
12. InterBase Server for Windows95 - Borland InterBase SQL серверининг кып фойдаланувчили версияси.

МБ билан ишловчи компонентлар

МБ иловаларини щосил =илувчи компонентларни кырамиз. Бу компонентлар щам бош=а компонентларга ыхшаб визуал ва новизуал компонентларга былинади.

МБ билан ишлаш имконини берувчи компонентлар Data Access, Data Controls, DataSnap, Desicion Cube, QReport ва InterBase компонентлар палитрасида жойлашган. Айрим компонентлар масофавий МБ билан ”клиент-сервер” архитектурасида ишлаш учун мылжалланган. +уйида Delphi5 учун МБ га мылжалланган компонентлар палитраси келтирилган.

Data Access сащифасида маълумотларга мурожаатни ташкил =илишга мылжалланган новизуал компонентлар жойлашган (20-расм):



20-расм. Data Access сащифасининг компонетлари.

Data Source - ахборот манбаси;
table – МБ жадвалига асосланган ахборот тыплами;
Query - SQL-сыровга асосланган ахборот тыплами;
StoredProc - сервернинг са=ланувчи процедурасини ча=ириш;
Database - МБ билан бо\ланиш
Session - МБ билан ишловчи жорий сеанс;
Batchmove - ёзувлар гурущи устида амал бажариш;
UpdateSQL – SQL-сыровга асосланган ахборот тыпламининг модификацияси;
NestedTable - ичма-ич жадвал;

Data Controls сащифасидаги маълумотларни бош=ариш учун мылжалланган визуал компонентлар жойлашган (21-расм):

21-расм. Data Controls сащифасининг компонентлари.

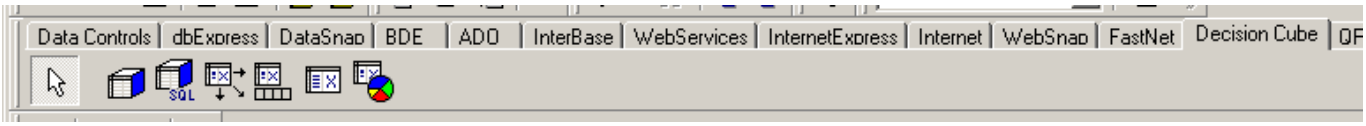
DBGrid - сетка (тыр);
DBNavigator - навигацияли интерфейс;
Dbtext – ёзув;
DBEdit - 1 =аторли мущаррир;
DBMemo - кып =аторли мущаррир;
DBImage - график тасвир;
DBListbox - оддий рыйхат;
DBCombobox - комбинациялашган рыйхат;
DBCheckbox - муста=ил ытказгич;
DBRadioGroup - бо\ли= ытказгичлар гурущи;
DBLookupListBox - бош=а ахборотлар тыпламининг майдони быйича шаклланадиган оддий рыйхат;
DBLockupComboBox - бош=а ахборотлар тыпламининг майдони быйича шаклланадиган комбинациялашган рыйхат;
DBRichEdit - тыла функционал матн мухаррири;
DBCtrlGrid - модификациялашган тыр;
DBChart - диаграмма;

DataSnap сащифасида масофавий МБ ни бош=ариш учун мылжалланган компонентлар жойлашган (22-расм):

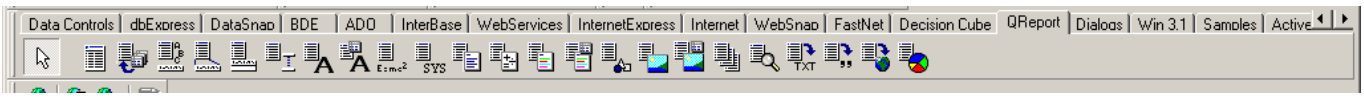
22-расм. Data Snap сащифасининг компонентлари.

ClientDataSet - ахборотни клиент даражасида киритиш;
DCOMConnection - кып о=имли иловалар учун масофавий сервер билан бо\ланиш;
SocketConnection - Windows сокети ор=али масофавий сервер билан бо\ланиш;
DataSetProvider - ахборотни киритиш провайдери;
SimpleObjectBroker - оддий объект брокери;
WebConnection – web-сервер билан бо\ланиш;
CorbaConnection - CORBA-клиентни улаш.

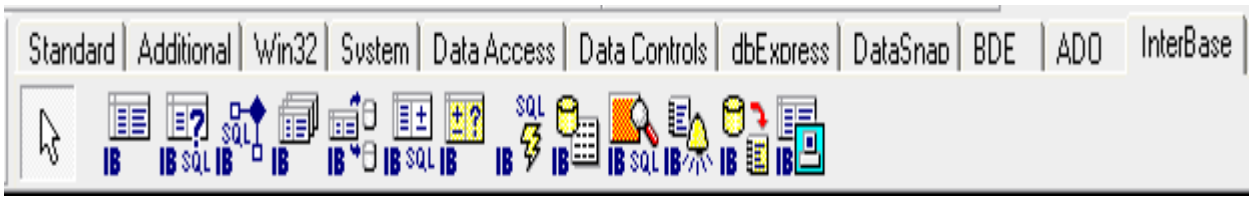
Desicion Cube сащифасининг компонентлари:



Qreport сащифасининг компонентлари:



InterBase сащифасининг компонентлари:



DataBase Desktop ёрдамида МБ ни щосил =илиш

DataBase Desktop одатда, Delphiнинг асосий менюсидаги Tools былимида келтирилган былади. Агар у ерда былмаса, у щолда Tools | Configure Tools командаси ор=али DataBaseDesktopни ча=ириш мумкин. У ча=ирилгач, унинг ойнаси экранга чи=ади. Энди Paradox7 МББТ ёрдамида МБ жадвалини щосил =илишни кырамиз. Paradox7 да МБ бу алощида каталог былиб, унда кенгайтмаси .db былган файллар-жадваллар жойлашади. Шунинг учун янги каталог очиб =ыйиш керак былади (масалан, ”проводник” дастури ёрдамида). Сынгра File| New командаси берилади. Бунда 3 хил вариантли =исм меню чи=ади:

QBE Query ...	Сыровларни визуал щосил =илиш ва уларни файлга ёзиш
SQL File	Сыровларни SQL тилида щосил =илиш ва уни файлга ёзиш
Table	Янги жадвал щосил =илиш

Бу ердан Tablени танлаш керак. Бу щолда кичикро= диалог ойнаси очилади. Ундан Paradox7 ни танлаймиз. Сынгра щосил былган янги ойнадан биз жадвалимизнинг тузилишини, яъни ундаги майдонлар ва уларнинг тоифаларини ва бош=а кыпгина ахборотни киритишимиз лозим былади.

Майдонларни бош=ариш

Щар бир щосил =илинадиган жадвал учун майдонлар номи, яъни идентификатори ани=ланиши лозим. Майдон номи 25 та символгача былиши ва биринчи символи пробел былмаслиги керак. Сынгра бу майдоннинг тоифаси ани=ланади. Бунинг учун Type былимига ытилади ва сич=ончанинг ынг тугмаси босилади. Майдон тоифаси сифатида =уйидагилардан бирини бериш мумкин:

Белгиланишти	Рыйхатда берилиши	Тавсифи
A	Alpha	1 дан 24 гача былган ихтиёрий ASCII символларидан ташкил топган матнавий майдон
N	Number	-10^{307}-10^{308} былган ща=и=ий сонлар
$	Money	Пул бирлиги кыринишидаги мусбат ёки манфий сонлар
S	Short	-32 767 ÷ + 32 767 гача былган бутун сонлар
I	Long Integer	-2 147 483 648 ÷ +2 147 483 647 гача былган бутун сонлар
#	BCD	BCD (Binary Coded Decimal) формуласидаги сонлар
D	Date	Сана кыринишидаги =ийматлар
T	Time	Ва=т кыринишидаги =ийматлар
@	Timestamp	Ва=т ва санани са=ловчи =ийматлар
M	Memo	Чегараланмаган узунликдаги матнларни са=лайди
F	Formatted memo	Чегараланган узунликдаги форматланган матнларни са=лайди
G	Graphic	.bmp, .pex, .tif, .gif ёки .eps форматли файллардаги расмларни .вмр форматга

		кычиради
O	OLE	OLE тоифасидаги маълумотлар. Бу тасвирлар, товушлар, щужжатлар
L	Logical	Манти=ий майдон. True ёки false =ийматига эга
T	Autoincrement	Автоматик равишда 1 бутун сонга оширади
B	Binary	.mb файллардаги иккилик ахборотни са=лайди. Буларда товуш ёки ихтиёрий бош=а маълумотлар былиши мумкин
Y	Bytes	Binary дан фар=ли томони файлларда эмас жадвалларда са=лайди.

Калит майдонлар энг охирги устунида «*» белгиси билан тугаши лозим.

Жадвал хоссаларини ани=лаш

Table propertiesда жадвал хоссалари ани=ланади. Ундан =уйидагиларни танлаш мумкин:
1.Validity cheks -=ийматларнинг ты\рилигини текшириш.

Унда майдонни =уйидаги характеристикаларини бериш мумкин:

required field	бу индикаторда майдоннинг щар бир ёзуви бор былган майдонлар белгиланади
Minimum	Минимал =иймат. Бу хоссани сонли майдонларга бериш фойдали
Maximum	Максимал =иймат. Бу хоссани сонли майдонларга бериш фойдали.
defaulf	Жамлик быйича =иймат. Бу хоссани сонли ва манти=ий майдонлар учун бериш фойдали
picture	Ахборотни киритиш учун шаблон. Бу хоссани телефон номер учун бериш мумкин
assist	Бу тугмача picture шаблонини щосил =илишга ёрдам берувчи диалог ойнани ча=иради

2. Table Lookup - кыриш жадвали.

Бу былим жадвалнинг майдонини бош=а жадвалнинг =андайдир майдони билан бо\лайди.

3. Secondary Indexes - иккиламчи индекслар.

Бу былим кейинги ишларда осон былиши учун иккиламчи индекслар щосил =илади. Янги иккиламчи индекс щосил =илиш учун Define-тугмаси босилади. Сынг indexed fields ойнасидан майдонлар рыйхати танланади.

Индекс шаклланган, ОК тугмачаси босилади ва янги ойнага индекснинг номи киритилади.

4. Referential Integrity - ишоралар даражасидаги яхлитлик.

Адабиётлар рўйхати

1. А.Я.Архангельский. Программирование в Delphi6.-М.: ЗАО «Издательство БИНОМ», 2001.
2. П.Г.Дарахвелидзе, Е.П.Марков, О.А.Котенок. Программирование в Delphi5.-СПб.: БХВ-Петербург, 2001.
3. С.Бобровский. Delphi5. Учебный курс-СПб.: Издательство «Питер», 2000.
4. Н.Тюкачев, Ю.Свиридов. Создание мультимедийных проектов, 2001.
5. В.Э.Гофман, А.Д.Хомоненко. Работа с базами данных в Dephi. - СПб.: БХВ-Петербург, 2001.

Мундарижа:

Мавзу номи **Бет**