11- Mavzu: DELPHI obyektga yo’naltirilgan dasturlash tili

Delphi dasturlash muhiti

Delphi - Windows operatsion tizimida dastur yaratishga yo’naltirilgan dasturlash muhitidir. Delphida dastur tuzish zamonaviy vizual loyihalash texnologiyalariga asoslangan bo’lib, unda dasturlashning obyektga mo’ljallangan g’oyasi mujassamlashgan. Delphida dastur Turbo Pascal dasturlash tilining rivoji bo’lgan Object Pascal tilida yoziladi.

Delphi - bir necha muhim ahamiyatga ega bo’lgan texnologiyalar kombinatsiyasini o’zida mujassam etgan:

• yuqori darajadagi mashinali kodda tuzilgan kompilyator;

• obyektga mo’ljallangan komponentalar modullari;

• dastur ilovalarini vizual tuzish;

• ma’lumotlar bazasini tuzish uchun yuqori masshtabli vosita.

Delphi - Windows muhitida ishlaydigan dastur tuzish uchun qulay bo’lgan vosita bo’lib, kompyuterda dastur yaratish ishlarini avto- matlashtiradi, xatoliklarni kamaytiradi va dastur tuzuvchi mehnatini yengillashtiradi. Delphida dastur zamonaviy vizual loyihalash texnolo- giyasi asosida obyektga mo’ljallangan dasturlash nazariyasini hisobga olgan holda tuziladi. Delphi tizimi Turbo Pascal 7.0. tilining rivoji bo’lgan obyektga mo’ljallangan Object Pascal dasturlash tilini ishlatadi.

Ma’lumki, dastur tuzish sermashaqqat jarayon, lekin Delphi tizimi bu ishni sezilarli darajada soddalashtiradi va masala turiga qarab dastur tuzuvchi ishining 50—80%ni tizimga yuklaydi. Delphi tizimi dasturni loyihalash va yaratish vaqtini kamaytiradi, hamda Windows muhitida ishlovchi dastur ilovalarini tuzish jarayonini osonlashtiradi.

Delphi o‘zida bir qancha zamonaviy ma’lumotlar bazasini boshqarish tizimlarini va dasturlash texnologiyalarini ma’lumotlar bazasini yaratishda ishlatadi.

Delphi tizimining oynasi va uning elementlari

• bosh oyna — Delphi Projectl;

• forma oynasi — Forml;

• obyekt xossalarini tahrirlash oynasi —Object Inspector;

• obyektlar ro’yxatini ko’rish oynasi — Object tree View;

• dastur kodlarini tahrirlash oynasi — Unit.pas.

Bosh oyna ekranning yuqori qismida joylashgan bo’lib, uning birinchi qatorida sarlavha, ya’ni, proyektning nomi joylashgan. Ikkinchi qatorda buyruqlar menyusi

gorizontal ko’rinishda joylashgan. Keyingi qatorning chap tarafida uskunalar paneli va o’ng tarafida komponentalar politrasi joylashgan.
Buyruqlar menyusi quyidagilarni o’z ichiga olgan:
— File (fayl) bo’limi fayllar ustida ish bajarish uchun kerakli buyruqlarni

o‘z ichiga olgan;

— Edit (tahrir) bo‘limi fayl ichidagi ma’lumotlarni tahrirlash uchun kerakli buyruqlarni o‘z ichiga olgan:

— Seerch;

— View;

— Compile;

— Run formani ishga tushirish;

— Options;

— Tols servis xizmatidan foydalanish;

— Help yordam chaqirish.

Object Inspector

Object Inspector oynasi obyekt xossalarini tahrirlash uchun xizmat qiladi. Obyekt xossalari bu — obyektga berilgan xarakteristika bo’lib, uning ko’rinishi, joylashishi va holatidir. Masalan, Width va Height xossalari forma o’lchamini, top va Lift esa formaning ekrandagi holati, Caption — sarlavha matnini aniqlaydi.
Vizual dasturlash texnologiyasida obyekt deganda muloqot oynasi va boshqarish elementlari (kiritish va chiqarish maydoni, buyruq tugmalari, pereklyuchatellar va boshqa) tushuniladi.
Delphida dasturlash ikkita o’zaro ta’sir etuvchi bir-biri bilan bog’liq jarayon asosida tashkil qilinadi:
— dasturni vizual loyihalash jarayoni;

— dastur kodlarini kiritish (yozish) jarayoni.

Kodlarni yozish uchun maxsus kod oynasi mavjud bo’lib, u dastur matnini kiritish va tahrirlash uchun mo’ljallangandir. Bu kodlarni yozish oynasida dasturlash Pascal tilining rivoji bo’lgan va kengaytirilgan Object Pascal tilida tuziladi.
Kodlarni yozish oynasi boshlanishda o‘z ichiga holi bo‘sh formani akslantiruvchi dastur matnini yozib chiqaradi. Dastur loyihasini ishlash mobaynida dasturchi kerakli dastur operatorlarini kiritib, formani loyiha bo’yicha akslantiradi. Delphida dasturlash forma oynasini tashkil etishdan boshlanadi.
Oddiy dastur ilovasini yaratish ketma-ket **File=> New=> Application** buyrug’ini berish bilan boshlanadi. Bu buyruqni berishdan oldin ikkita asosiy ishni bajarish lozim:

— papka tashkil etish;

— tizimni to’g’rilash.

Kengaytmali fayllar

Delphi dasturlash muhitida ishlash jarayonida quyidagi kengaytmali fayllar ishlatiladi:

— loyiha fayli, kengaytmasi **.dpr**

— paskal moduli fayli, kengaytmasi **.pas**

— komponentalar joylashgan fayl, kengaytmasi **.dcu**.

— formalar joylashgan fayl, kengaytmasi **.dfm**;

— ma’lumotlar bazasi fayli, kengaytmasi **.dbf**.

Tayyorlanadigan Delphi dastur uchta asosiy etapdan o‘tadi:

— kompilyatsiya;

— komponovka;

— bajarish.

Kompilyatsiya

Kompilyatsiya etapida tayyorlangan dastur matni Object Pascal tiliga o‘tkaziladi. Kompanovka bosqichida esa kerakli qo‘shimcha yordamchi dasturlar va ostdasturlar unga birlashtiriladi. **F9** tugmasini bosish bilan **Save UnitAs** dialog oynasi paydo bo‘ladi va sizdan **Unit.pas** moduli uchun fayl nomini va joylashadigan papkani ko‘rsatishingizni so‘raydi. Agar joyi ko‘rsatilmasa Delphi avtomatik ravishda dasturingizni **Bin** papkasiga joylashtiradi. Yaxshisi siz bu papkani o‘z ishchi papkangiz nomiga almashtiring, masalan, **My_Delphi**. Dastur kompilyatsiya qilinishi paytida Delphi sistemasi **pas**, **dfm** va **dcu** kengaytmali modullar tuzadi. **.pas** kengaytmali fayl kodlarni yozish oynasiga kiritilgan dastur matnini, **.dfm** forma oynasi tashkil etuvchilarini, **.dcu** kengaytmali fayl esa **.pas** va **.dfm** kengaytmali fayllarning birgalikdagi mashina kodiga o‘tkazilgan variantini saqlaydi. Bu **.dcu** kengaytmali fayl kompilyator tomonidan tashkil qilinadi va yagona ishchi (bajariluvchi) **.exe** kengaytmali fayl tashkil qilishga baza yaratadi.

Delphi loyihasining tuzilmasi

Delphi dasturi — bu bir necha bir-biri bilan bog‘liq fayllardir. Har qanday dastur . dpr kengaytmali loyiha fayli va bir yoki bir necha . pas kengaytmali modullardan tashkil topadi. Loyiha fayli dasturchi tomonidan kiritilmaydi, u foydalanuvchining ko‘rsatmalari asosida avtomatik ravishda Delphi sistemali dasturi tomonidan tuziladi. Loyiha fayli matnini ko‘rish uchun Project/View Source buyrug‘ini berishi zarur. Loyiha matni umumiy holda quyidagicha bo‘lishi mumkin:

Program Projectl;
Uses
Forms,
Unitl in 'Unitl.pas' {Forml}
{$R *.res}
Begin
Application.Initialize;
Application.CreateForm(Tform1, Forml);
Application.Run;
End.

Loyiha nomi dasturchi tomonidan loyiha faylini saqlash vaqtida beriladi va u Delphi muhitida bajariluvchi fayl, ya'ni, kengaytmasi .exe bo'lgan faylni tashkil qilishni aniqlaydi. Loyiha faylidan keyin ishlatiladigan modullar: standart modullar Forms va Unitl joylashadi. **{$R *.res}** direktivasi kompilyatorga ishlatilishi kerak bo'lgan resurs fayllari, masalan dasturlarni e'lon qilish kerakligini bildiradi. Yulduzcha belgisi resurs faylining kengaytmasi .res ekanligini bildirad

O'zgaruvchilar va konstantalar

O'zgaruvchilar. O'zgaruvchi nomi ostiga chizish belgisi yoki lotin harfidan boshlanuvchi lotin harflari, arab raqamlari va ostiga chizish belgilari ketma ketligi ya'ni identifikatordir. O'zgaruvchilarning quyidagi tiplari mavjuddir: **char** (simvol), **short** (qisqa butun), **int** (butun), **long** (uzun butun), **float** (haqiqiy), **double** (ikkilangan haqiqiy).

Butun sonlar ta'riflanganda qurilgan tiplar oldiga **unsigned** (ishorasiz) ta'rifi qo'shilishi mumkin. Ishorali ya'ni **signed** tipidagi sonlarning eng katta razryadi son ishorasini ko'rsatish uchun ishlatilsa **unsigned** (ishorasiz) tipdagi sonlarda bu razryad sonni tasvirlash uchun ishlatiladi

O'zgaruvchilar ta'rifi sodda shakli:

<tip> <o'zgaruvchilar_nomlari_ro'yxati >;

o'zgaruvchilarni ta'riflashda boshlangich qiymatlarini ko'rsatish mumkin.

Ko'rsatkichlar va ilovalar

Ko'rsatkichlar ta'rifi. Ko'rsatkichlar qiymati konkret tipdagi obyektlar uchun xotirada ajratilgan adreslarga tengdir. Shuning uchun ko'rsatkichlar ta'riflanganda ularning adreslarini ko'rsatish shart. O'zgaruvchi ko'rsatkichlar quyidagicha ta'riflanadi.

<tip> * <ko'rsatkich nomi>

Misol uchun **int * lp, lk** .

Ko'rsatkichlarni ta'riflaganda insializatsiya qilish mumkindir. Initsializatsiya quyidagi shaklda amalga oshiriladi:

Ko'rsatkichlarga o'xshab Ilovalarning qiymatlari ham adreslardir. Lekin Ilovalarning qiymatlarini o'zgartirish mumkin emas va Ilovalarga murojaat qilinganda avtomatik ravishda * qiymat olish amali bajariladi.

Ilovalar bilan ishlash qoidalari

Ilova o‘zgaruvchi emasdir. Ilovaga bir marta qiymat bergandan so‘ng uni o‘zgartirish mumkin emas. Bundan tashqari ilovalar ustida quyidagi amallarni bajarish mumkin emasdir:
Ilovaga ko‘rsatkich qiymatini berish mumkin emas. Ilovalarni solishtirish mumkin emas. Ilovalar ustida arifmetik amallar bajarish mumkin emas. Ilovani o‘zgartirish mumkin emas.

Foydalanuvchi funksiyalari

Funksiya ta’rifi. Funksiyani quyidagi ikki sifatda qarash mumkin:
hosila tiplardan biri;

dastur bajariluvchi minimal moduli.

Funksiya ta’rifi umumiy ko‘rinishi quyidagichadir:
<tip> <funksiya nomi>(<formal_parametrlar_ta’rifi>)
Formal parametrlarga ta’rif berilganda ularninga boshlangich qiymatlari ham kursatilishi mumkin.
Funksiya qaytaruvchi ifoda qiymati funksiya tanasida return <ifoda> ; operatori orqali ko‘rsatiladi.

Prototip

Agar programmada funksiya ta’rifi murojaatdan keyin berilsa, yoki funksiya boshqa faylda joylashgan bo‘lsa, murojjatdan oldin shu funksiyaning prototipi joylashgan bo‘lishi kerak. Prototip funksiya nomi va formal parametrlar tiplaridan iborat bo‘ladi. Formal parametrlar nomlarini berish shart emas.
Misol uchun:
float min(float, float);

Protseduralar

Protseduralar. Funksiyaga parametrlar qiymat bo‘yicha uzatiladi. Funksiyaga parametrlar qiymatlari uzatilishi haqiqiy parametrlar qiymatlarini funksiya tanasida o‘zgartirish imkonini bermaydi. Bu muammoni hal qilish uchun ko‘rsatkichlardan foydalanish mumkin.
Funksiyalarni qo‘shimcha yuklash. Funksiyalarni qo‘shimcha yuklashdan maqsad bir xil nomli funksiyaga har xil tipli o‘zgaruvchilar bilan murojaat qilib qiymat olishdir. Kompilyator haqiqiy parametrlar ro‘yxati va funksiya chaqirig‘i asosida qaysi funksiyani chaqirish kerakligini o‘zi aniqlaydi.

StringBuffer

StringBuffer — sinfida satrlar bilan ishlash uchun kerak bo‘lgan usullar mavjuddir. String sinfi obyektlari ma’lum uzunlikdagi o‘zgarmas satrlardan iborat. StringBuffer obyektlari uzunligii va qiymati o‘zgaruvchan satrlardan iborat. Java tilida ikkala sinfdan aktiv foydalaniladi, lekin dasturchilar + operatoridan foydalanilgan holda String sinfi obyektlarini ishlatadilar.

Konstruktorlar

StringBuffer sinfi obyektini parametrlarsiz yaratish mumkin, bu holda 16 simvol uchun joy ajratiladi va satr uzunligini o‘zgartirib bo‘lmaydi. Bundan tashqari konstruktorga bufer uzunligini ko‘rsatuvchi butun son berish mumkin. Bundan tashqari konstruktorga satr uzatish mumkin. Bu holda satrdan obyektga nusxa olinadi va qo‘shimcha 16 simvolga joy ajratiladi. StringBuffer obyekti uzunligini aniqlash uchun length usulidan, satr uchun ajratilgan joyni aniqlash uchun capacity usulidan foydalanish mumkin.

Tuzilmalar va funksiyalar

Ma’lumotlar turlarini Delphi tilida umumiy holda ikkiga ajratish mumkin: standart turlar. Bu turlar oldindan Delphi tili tomonidan aniqlangan bo‘ladi;

dasturchi tomonidan kiritiladigan (aniqlanadigan) turlar.

Standart turlar tarkibiga quyidagilar kiradi: butun, haqiqiy, belgili (simvol), qator (strok), mantiqiy, ko‘rsatgichli va variant.
Dasturchi turlarni dasturning **Var** bo‘limida o‘zgaruvchilarni tavsiflashda aniqlaydi yoki maxsus turlarni aniqlash uchun bo‘lim bo‘lgan -turlarni tavsiflash **Type** bo‘limida aniqlaydi.

Boshqaruvchi tugmalari

Sodda va eng ko’p ishlatiladigan tugmalar **Button va BitBtn** lardir. Bu kompanentalarning ko’p hususiyat, xodisa va metodlar bir xil. Asosiy farqlardan biri esa, **BitBtn** kompanentasida rasm qo’yish mumkinligidadir. Tugmalarning asosiy husussiyatlaridan biri **Caption** (sarlavha). **Caption** hususiyati ma’lum harfdan oldin qo’yilgan <&> belgisi orqali tugmaga tezda murojat qilish mumkin.. Misol uchun **Caption** xususiyatida <& Chiqish> yozilgan bo’lsin. Bu Formada <chiqish > shaklida ko’rinadi. Bu tugmaga murojat qilish uchun **Alt-C** tugmalarini bosishlik kifoya.

OnClick

Xar qanday tugmaning asosiy hodisasi **OnClick** bo’lib, bu hodisa tugma bosilganda sodir bo’ladi. Tugma bosilganda nima ish bajarilishi kerakligi aynan shu hodisada keltiriladi. Bundan tashqari sichqoncha va klaviatura orqali bo’ladigan bir qancha hodisalar mavjud. Bularni keyingi mavzularda o’rgani chiqamiz. Agar tugmani **Action** xususiyati bilan bo’lmagan bo’lsangiz **OnClick** xodisasida nima vazifa bajarilishi kerakligini yozish lozim.

Cancel

Cancel xususiyatida **true** qiymat o’rnatilgan bo’lsa, foydalanuvchi **Esc** tugmasini bosishi, tugmani bosishi bilan ekvivalent ishlaydi. Yani yugmani OnClick hodisasi bajariladi. Bu xodisani, turli dialog darchalarida, dialogni bekor qilish tugmalari ishlatish mumkin. Foydalanuvchi dialogni tugatish

uchun **Esc** tugmasini bosishi mumkin.
Default xususiyatida **true** qiymat o'rnatilgan bo'lsa, foydalanuvchi **Enter** tugmasini bosishi bilan ekvivalent ishlaydi. Agar bir nechta tugmada **Default** xususiyatida **true** qiymat o'rnatilgan bo'lsa, **TabOrder hususiyatidagi tartib bo'yicha bajariladi.**

Tugma programma kodi orqali murojat qilish

Tugma programma kodi orqali ham murojat qilish mumkin.
Formaning istalgan qismida <Ha> yoki <h> tugmalarini bosishlik bilan hissoblash jarayonini amalgam oshirmoqchisiz. Buning uchun formaning **KeyPreview** husussiyati **true** qilinadi
Rasm va sarlovhaning tugmada joylashishi **Margin, Layout, Spadeg** hususisyatlari orqali beriladi. Agar **Margin** =-1 bo'lsa, rams va sarlovha tugma markazida bo'ladi. Bu holatda rasmni sarlovhaga nisbatan qayerda turishi **Layout** hususiyati orqali aniqlanadi. **BlGlyphTop** (tepada), **BlGlyphButtom** (pastdan) **BlGlyphLeft (**chapda**) BlGlyphRight (**o'ngda**).** Agar **Margin > 0** bo'lsa, Layout qiymatiga tugmaning qarab u yoki bu chegarasidan **Margin** da berilgan pikselcha joy tashlanadi.

SpeedButton tugmasi

SpeedButton tugmasidan oddiy boshqaruv tugmalar kabi foydalanish mumkin. Bundan tashqari **SpeedButton** tugmasidan fiksirlangan tugma sifatida foydalanish mumkin. Bu tugmalar odatta vazifalar panelida, menyular qatoridagi ko'po ishlatiladigan buyuruqlar nushasini ko'rsatish uchun ishltiladi.
SpeedButton tugmasida boshqa tugmalar kabi **Caption** hususiyatlari mavjud. Lekin bu odatta bo'sh bo'ladi, uning o'rniga rasm (piktogramma) ishlatiladi. Tugmaga rasm Glyph hususiyati orqali beriladi **NumGlyph, Layout, Margin, Spacing** hususiyatlari **SpeedButton** tugmasi uchun ham o'rinli.
SpeedButton tugmasining sosiy husussiyatlaridan biri **GroupIndex (**guruhlar indeksi**).** Agar **GroupIndex=**0 bo'lsa, tugma huddi **Button, Bitbtr** tugmalari kabi qachonki qo'yib yuborsa, tugma o'z holiga qaytadi.
Agar **GroupIndex>0 va AllowAllUp=true** bo'lsa, foydalanuvchi tomonidan tugma bosilganda, tugma bosilib qoladi. Qachonki tugma ikkinchi marotaba bosilganda o'z holiga keladi. (ikkinchi marotaba bosganda tugma o'z holiga kelishi uchun **AllowAllUp=true** bo'lishi kerak). **Down** husussiyati qaysi tugma bosilganini bildiradi. Yani qaysi tugma bosilsa , shu tugmaning **Down=true** ga o'zgaradi.
Programmani ishlab chiqish jarayonida **Down=true** qilingan bo'lsa programma ishga tushganda tugma bosilgan holda bo'ladi.
SpeedButton tugmasining **Flat** hususiyatini **true** ga o'zgartirish orqali chiroyli interfeys hosil qilish mumkin. Kursor tugma ustidan tushganda tugma o'z holiga qaytadi.
Barcha boshqariluvchi kompanentalarning **Hint** hususiyati orqali ko'rsatma

satrni berish mumkin. Ko’rsatma satri kursor kompanenta ustiga kelganda, bu kompanenta nima ish bajarishi haqida ma’lumot beradi. Ko’rsatma satri formada ko’rinishi uchun kompanentaning **ShowHind** hyususisyati **True** qilinadi.

Memo va RichEdit kompanentalari

Memo va RichEdit kompanentalri ko’p satrli matnlarni tahrirlash uchun ishlatiladi. Barcha taxrirlash darchalaridagi kabi **Memo** va **RichEdit** kompanentalarida ham nusha olish **Ctrl-C (copy),** qirqib olish **Ctrl-X,** qo’yish **Ctrl-V, (paste),** oxirgi amalni bekor **(Ctrl-Z**), qilish imkoniyatlarini keltirgan. **Memo** kompanentasida format (Shrift, atributlar) barcha matn uchun bir xil bo’ladi va **Font** hususiyati orqali belgilanadi. Agar siz matnni faylga saqlasangiz, faqat simbollarni o’zida saqlovchi matnli fayl hosil bo’ladi. Bunda farmat saqlanmaydi. Saqlangan faylni **Memo** kompanentasiga ochganda **Font** hususiyatida o’rnatilgan shirift bilan ochiladi. Saqlanishdagi shirift bilan emas. **RichEdit** kompanentasi orqali RTF kengaytmali fayllar bilan ishlash mumkin. Shiriftni **SelAtributes** hususiyati orqali hohlagancha o’rnatish mumkin. Bu hususiyat **TTexAtributes** toifasida bo’lib, quyidagi ost hususiyatlarni o’z ichiga oladi: **Color** (rang**), Name** (shirift nomi), **Size** (o’lchami), **Style** (shakli) va boshqalar.
TabCount va **Tab hususiyatlari Want Tabs** hususiyati true bo’lganida manoga ega. Agar **Want Tabs=false** bo’lsa**,** foydalanuvchi **Tab** tugmasini bosganida **Fokus** keyingi kompanentaga uzatiladi. Biz Memo va RichEdit kompanentalrining asosiy farqlarini qirqib chiqdik.

Aligment va WordWrap

Aligment va **WordWrap** hususiyatlari matnlarni tekslash va uzun satrlarni keyingi satrga o’tkazish uchun ishlatiladi. **ReanOnly** hususiyatini **True** qilish irqali matnni o’qish mumkin. **MaxLength** hususiyati kiritilishi mumkin bo’lgan matn uzunligini bildiradi. Boshqa hollarda kiritilishi mumkin bo’lgan belgilar sonini bildiradi. **ScrollBars** hususiyati orqali kompanentaga siljitish yo’lakchasini o’mahs mumkin. **ScrollBars** hususiyati **ssNone** (siljitish yo’lakchasi o’rnatilmasin), **ssHorizantal** –(garizantal), **ssVertical** (vertikal**), ssBorth** (ham garizantal, ham vertikal) qiymatlarini qabul qilish mumkin. **Memo** va **RichEdit** kompanentalarining asosiy hususiyati **Lines. Lines** hususiyati **TSTrings** toifasiga tegishli bo’lib matnni satrlar ro’yhati sifatida saqlaydi.
Matnni programma ishlab chiqarish jarayonida ham kiritish mumkin. Buning uchun **Object Inspektor** dan **Lines** hususiyati to’g’risidagi uch nuqta bosiladi.

Text

Text hususiyati butun matnni o’zida saqlaydi. Matnning ma’lum satrida **Strings [Index:Inreger**] hususiyati orqali murojat qilish mumkin. **Index**

Delphi da 0 dan boshlanadi. Demak, **RichEdit1.Lines.Stringes [0]** matnni birinchi satri**. Count** hususiyati mantdagi satrlar sonini aniqlash uchun ishlatiladi. Taxrirlash darchasini tozalash uchun **Clear** protsedurasiga murojat qilinadi.
Matn oxiriga yangi satr qo'yish uchun, **Lines** hususiyatining **Add** yoki saqlash uchun esa **SaveTofile** metodlarig murojat qilinadi. Misol uchun ixtiyoriy tugmaga quyidagilarni kiritish mumkin.

Formalar

Ihtiyoriy ilovaning elementi forma (**konteyner**) hissoblanadi. **Formaga** boshqa ko'rinadigan va ko'rinmaydigan kompanentalarning joylashtirish mumkin**. Forna** foydalanuvchi nuqtai nazardan u ilova bilan ishlayotgan darchani. Ilovaga kiritlgan har bir yangi forma o'zining moduli (**Unit)**ga ega**. Modulda** forma bajarishi kerak bo'lgan funksiya protseduralar kiritiladi. Odatta murakkab ilovalar bir necha formadan iborat bo'ladi. Yangi ilova (**programma**) tuzish uchun file menyusidan new oplication komandasi tanlanadi. Odatda birinchi forma asosiy forma hisoblanadi. Ilovaga yangi forma qo'shish uchun file menyusidan new form tanlanadi.

Show va ShowModal metodlari

Bir formadan boshqasiga o'tish uchun Show va ShowModal metodlaridan foydalanish mumkin. ShowModal metodi joriy formani yopgandan keyin boshqa formalar bilan ishlashga ruxsat beradi. Bu metod operativ xotiradan unumli foydalanish uchun ishlatiladi.
Show va ShowModal metodlarini ayni vaqtda ko'rinmaydigan formalar uchun ishlatiladi. Agar forma ko'rinish ko'rinmasligi noma'lum bo'lsa, quyidagicha programma kodi keltiriladi.
If (not Form2.visibli)then Form2. ShowModal
Show va ShowModal metodlari bajarilganda formaning OnShow hodisasi sodir bo'ladi. Hide metodi orqali formani ko'rinmaydigan qilish mumkin. Formani close metodi orqali yopish mumkin. Bir necha forma bilan ishlaganda bir formadan boshqasiga o'tish uchun shift+F12 tugmasi bosiladi.

Foydalanilgan adabiyotlar

1. Informatika va informatsion texnologiyalar, M. Aripov va boshqalar. Oliy o’quv yurti talabalari uchun darslik. Toshkent-2019 y.
2. Axborot texnologiyalari, M. Aripov va boshqalar. Oliy o’quv yurti talabalari uchun o’quv qo’llanma. Toshkent-2019 y.
3. Delphi tilida dasturlash asoslari, Sh. Nazirov. Toshkent-2018 y.