

Қолданбалы программаларды құру
және қолдану

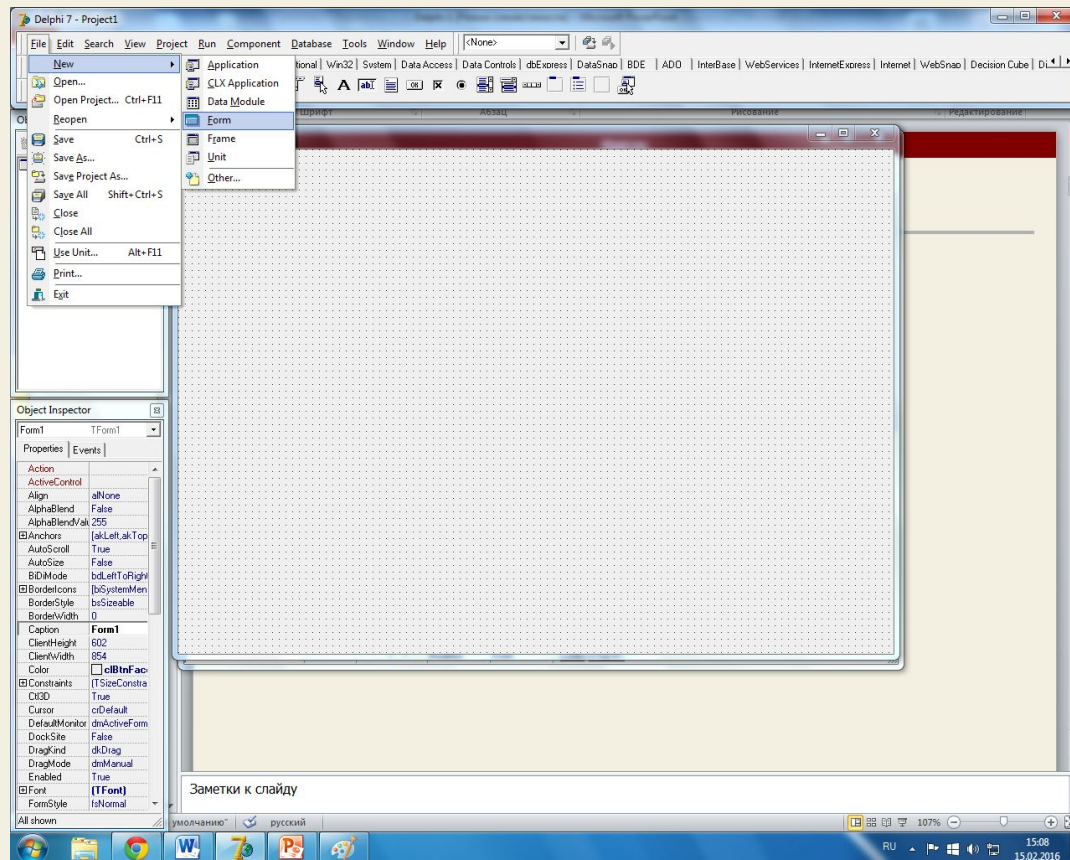
ҚОСЫМША ПІШІНДЕРМЕН ЖҰМЫС

Borland®

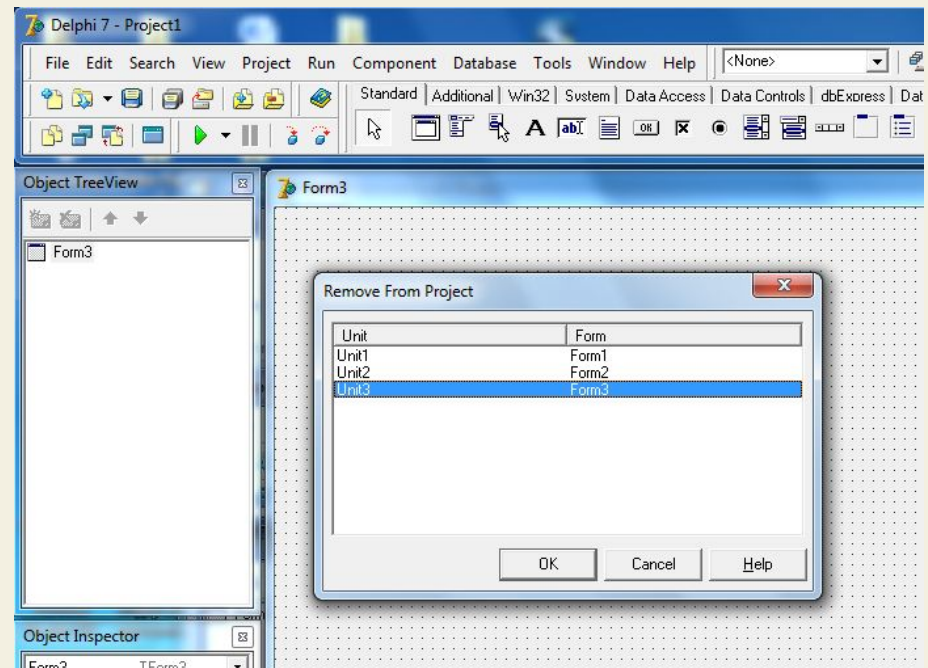
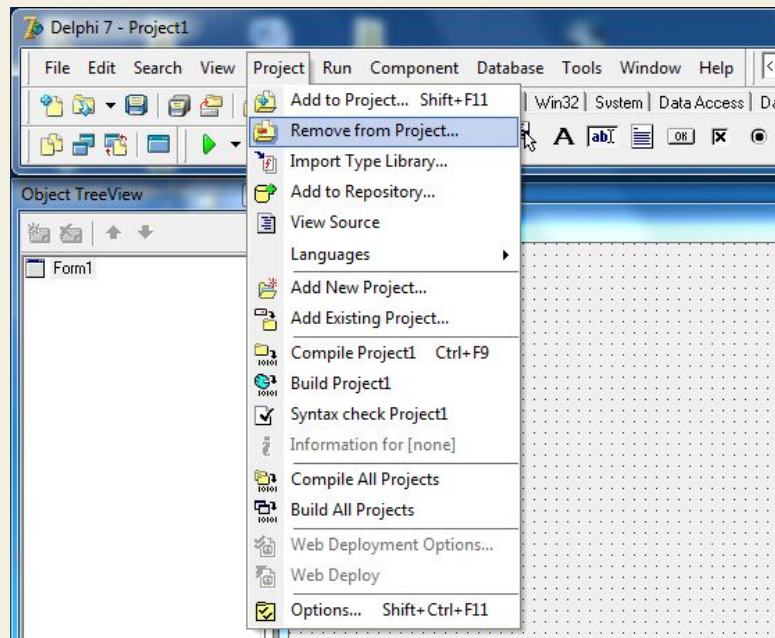
DELPHI 7

Delphi –де де қосымша Пішіндермен жұмыс істеуге мүмкіндік бар. Мұнда біз тұтынушымен сұхбат жүргізуге және қажетті ақпаратты қабылдауға және оны шығаруға мүмкіндік беретін қосымша пішіндерді оңай құра аламыз. Біз мұнда Delphi –дің негізгі пішініне қосымша бірнеше пішіндерді құрып үйренеміз.

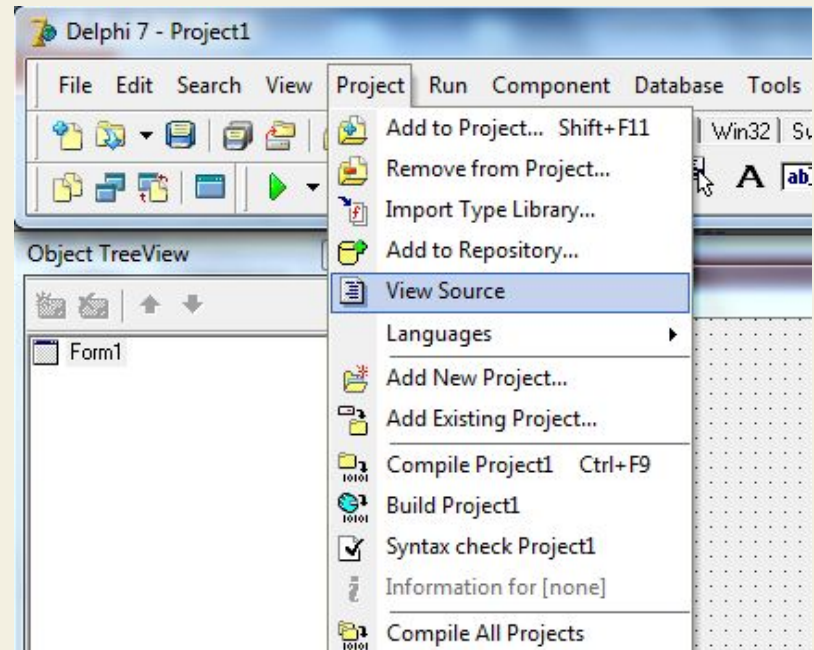
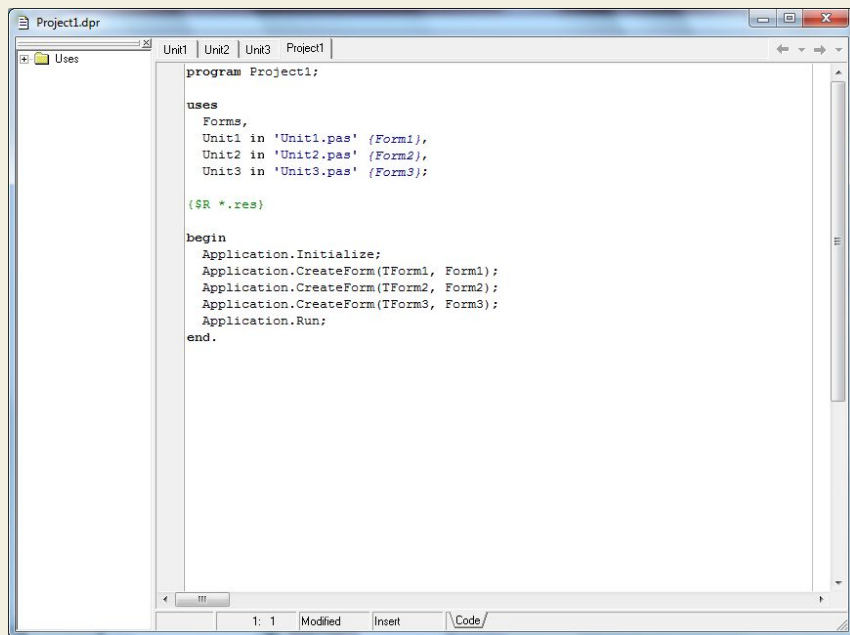
Бағдарламаға жаңа пішін енгізу басты беттегі батырманы (New Form) басу арқылы, соған қоса тиісті командаларды: **File -> New -> Form** орындау арқылы жүзеге асырылады.



Пішін оның жұмысын сипаттайтын жаңа модульмен бірге құрылады. Оны бағдарламадан жоюға болады: ол үшін батырма бар және мына меню командасы: **Project -» Remove from project....** қолданылады. Пішін модульмен бірге құрылатындықтан, көрінген терезеден жойылуға тиісті модульді таңдау керек.



Project -> View Source
командасын орындайық. Код редакторында (Бас терезенің коды бейнелетін пішіннің емес) жаңа астар пайда болады.



Мұнда Бас терезе көрінбейді, бірақ барлық жобаны басқарады және жалпы пішінсіз де жұмыс істей алады. Оған өз кодыңды орналастырып және Паскальдағы сияқты бағдарлама жазуымызға болады.

Project menu options:

- Add to Project... Shift+F11
- Remove from Project...
- Import Type Library...
- Add to Repository...
- View Sources
- Languages
- Add New Project...
- Add Existing Project...
- Compile Project1 Ctrl+F9
- Build Project1
- Syntax check Project1
- Information for [none]
- Compile All Projects
- Build All Projects
- Web Deployment Options...
- Web Deploy
- Options... Shift+Ctrl+F11

Project Options for Project1.exe dialog:

- Directories/Conditionals
- Version Info
- Packages
- Forms
- Application
- Compiler
- Compiler Messages
- Linker
- Main form: Form1
- Auto-create forms: Form1, Form2, Form3
- Available forms:
- Default
- OK
- Cancel
- Help

Object Inspector properties for Form1:

- Action
- ActiveControl
- Align: alNone
- AlphaBlend: False
- AlphaBlendValue: 255
- Anchor: [akLeft, akTop]
- AutoScroll: True
- AutoSize: False
- BiDiMode: bdLeftToRight
- BorderStyle: bsSystemMenu
- BorderStyle: bsSizeable
- BorderWidth: 0
- Caption: Form1
- ClientHeight: 602
- ClientWidth: 854
- Color: clBtnFace
- Constraints: (TSizeConstraint)
- Cursor
- DefaultMonitor
- DockSite
- DragKind: dkManual
- DragMode: dmManual
- Enabled: True
- Font: (TFont)
- FormStyle: fsNormal

Annotations:

- Пішіндермен жұмыс (Project menu)
- Пішіндермен жұмыс (Project menu)
- Бас пішін (Main form)
- Өздігінен жасалатын пішіндер (Auto-create forms)
- Қол жетімді пішін (Available forms)

Сонымен Form1 Бас пішіні өзі жасалады, ал қосымша Form2 пішінін біз бағдарламада қажетіне қарай өзіміз құрамыз. Егер біз осы айтылғанды жасамаған болсақ, онда экранда жаңа пішін шығару үшін былай деп жазсақ жеткілікті :

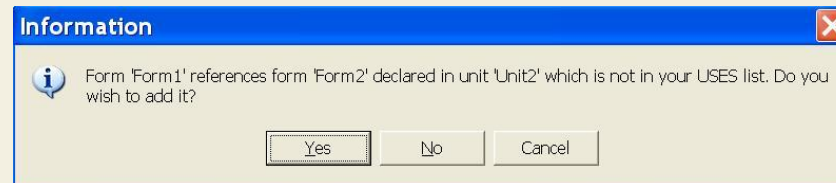
```
Form2.Show; // жай пішін үшін  
Form2.ShowModal; // модальды пішін үшін
```

Егер біз қосымша пішіндерді қолжетімді түрге көшірген болсақ, онда осындай пішінді шақыру алдында, оның бар-жоқтығын былайша тексеру қажет:

```
if (not Assigned(Form2)) then // Пішіннің  
бар-//жоқтығын тексеру  
    Form2:=Form2.Create(Self); // Пішінді құру  
    Form2.Show; // (немесе Form2.ShowModal) //Пішінді  
    //көрсету
```

Енді жай пішінмен Модалды пішіннің айырмашылығын қарастырайық. Жай пішін экранда орналасқан барлық пішіндердің бірінен біріне еркін көшуге мүмкіндік береді.

Ал модалдық пішін оны шақырған сәтте жобаның пішіндері арасындағы көшуді жабады да, жабылғанша тек сол ашылған пішінмен ғана жұмыс істейсіз.



Оған қарамастан екінші пішінді
шақыруға әрекет жасасаңыз,
бағдарлама мынадай сауал
шығарады

Form1 пішіні Unit 2 модулінде
жарияланған Form2 пішінін шақырып
тұр, бірақ ол пішін қолданылған
модульдер тізімінде жоқ.

Сіз оны қосқыңыз келе ме?

Мұны директива **{SR *.dfm}** алдындағы модульдің басына мына бағдарлама үзiгiн **uses Unit2;**

қосу арқылы шешу керек. Негізінде мұны компиляция алдында «қолмен» қосуға да болар еді. Сонда сауал да болмайды. Бірақ соның қажеті бар ма?

Мұнда «Yes»» деп жауап қайтарып және F9 – ды басамыз.

Алдымен пішінге оны жабу операциясын енгіземіз. Мұны бірнеше тәсілмен істеуге болады. Батырманы алып, оған «Закреть» деп жазып, сосын **OnClick** өңдеуішке былай жазамыз:

Form2.Close; // негізінде жай ғана Close;

Бұл оператор оны пішін менюінен шақырған кезде жұмыс істейді, әрине меню (StandardБұл оператор оны пішін менюінен шақырған кезде жұмыс істейді, әрине меню (Standard астарындағы MainMenu компоненті) оған енгізілген жағдайда . Бұл туралы алда әңгімелейтін боламыз.

Бізге енді модальды пішінге жататын пішіндердің жабылу тәсілдерін қарастыруымыз керек. Оның сұхбат жасауға арналған терезелерінде сұрақтарға жауап беруді қажет етеді.

Ол үшін пішінге мына : «Иә», «Жоқ» «Болдырмау» және т.б жауаптарға сәйкес батырмаларды орналастыру қажет. Әр батырманың **mrYes**, **mrNo**, **mrCansel** және басқа мәндерге ие **ModalResult** қасиеті бар. Мұнда таңдалған батырманың **ModalResult** мәні пішіннің осы қасиетіне беріледі. Бұл қасиет объектілер инспекторынан көруге болатын пішін қасиетінің тізімінде жоқ, бірақ оны бағдарламалық түрде таба аламыз. («Form2» деп жазып нүкте қой, сосын шыққан тізімнен ізде)

ModalResult қасиетінің **mrNone** мәнінен бөлек мәні бар батырмасын бассак (тіпті батырмада өңдеуші болмаса да) ол пішіннің жабылуына әкеледі. Осыдан кейін пішіннің осы қасиетін таңдау арқылы, тұтынушының қойылған сұраққа қандай жауап бергенін анықтауға болады:

procedure

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

begin

```
Form2.ShowModal;
```

```
if Form2.ModalResult=mrYes then //
```

```
Бұл оператор Form2 //жабылған соң  
ғана қолжетімді болады
```

```
Form1.Caption:='Тұтынушы оң  
жауап берді!';
```

```
end;
```



Осы мысалдан көріп отырғанымыздай, бір пішіннен басқа пішіннің қасиеттеріне , сол сияқты олардың компоненттеріне қатынас жасау үшін сол пішіннің аты көрсетілуі қажет екен. Оған қоса біз оның жұмысын сипаттайтын модульде қолданылатын мәліметтерге де қол жеткізе аламыз. Ол үшін де модульдің аты көрсетілуі қажет. Мысалы, **Unit2** модульіндегі X айнымалымына қатынас жасау үшін:

Unit2.X деп жазамыз.

2. AlfaBlend қасиеті (объектінің тұнықтығы)

Көбінесе бағдарламада пішінді жабу сәтінде белгілі бір операциялар орындалуы тиіс. Ол пішіннің **OnClose** оқиға өңдеуішінде жасалады. Ал енді пішінді жабуды болдырмауды қарастырайық. Осы мақсатта **OnCloseQuery** оқиға өңдеуішін қолдануға болады. Ол өңдеуіштегі **CanClose** логикалық айнымалысының қабылдайтын мәніне байланысты. Пішін **CanClose:=True;** мәнін қабылдағанда ғана жабылады.

Егер біз мынадай код жазсақ:



```
procedure TForm1.FormCloseQuery(Sender: TObject; var CanClose:  
Boolean);  
begin  
    CanClose:=False;  
end;
```

онда тұтынушы бағдарламаны тіпті де жаба алмайды, тек оны Windows-тың Міндеттер Диспетчерін/Диспетчер задач/ қолданып қана орындай алады.

Сабақ қорытындысы:

**Бұл сабақта біз Borland Delphi программалау жүйесінде
пішіндермен жұмыс істеуді қарастырдық.**
