1 VISUAL STUDIO.NET ВИЗУАЛДЫ БАҒДАРЛАМАЛАУ ОРТАСЫ

1.1 Объекті-бағытталған бағдарламалауға кіріспе

Оқу құралының алдыңғы бөлімінде деректер, әдістер, сонымен қатар C# тілінде жазылған бағдарламаның өзі де кластарда орналасуы керектігі жөнінде ескертілген.

Кластардың пайда болуы бағдарламалау технологиясын өзгертті. Егер бұрын құрылымдалған бағдарламалаудың негізгі бірлігі функциялар мен процедуралар болса, кластардың пайда болуы деректерді және сонымен қатар әдістерді біріктіретін бағдарламаның функцияналды аяқталған модульдерін құруға мүмкіндік берді. Осындай бағдарламалардың негізгі бірлігі - кластар (объекттер), кластар арқылы бағдарламалау технологиясы объекті-бағытталған технология деп аталады.

Кітаптың осы бөлімінде Windows қосымшаларды (күрделі бағдарламалық жүйелер) жобалау кезінде объекті-бағытталған технологияны қолдануға байланысты сұрақтар қарастырылады. Объекті-бағытталған бағдарламалау (ОББ) технологияларында кластардың қолданылуы олардың екі қызметті орындай алатынын көрсетеді: бағдарлама модулі немесе деректер типі ретінде қолданылуы.

Құрылымның модульдігі – Windows қосымшаларының негізгі қасиеті. Үлкен бағдарламалық жүйені модульдерге бөлмей дайындау кезінде бағдарламалаудың модульдік технологиясы арқылы дайындаған жүйеге қарағанда уақыт едәуір көп жұмсалады. ОББ-да Windows қосымшалар модульдік принципте әзірленеді, олар модульдің негізі болатын кластардан тұрады. Құрылымның модульдігі – күрделі бағдарламалық жүйелерді әзірлеу процессін жылдамдату бойынша негізгі құрал.

Екінші жағынан класс деректер типі болып келеді. Windows қосымшаларды объекті-бағытталған жолмен әзірлеу деректерге сүйеніп жобалау деп аталатын стильге негізделген. «Жүйелерді жобалау нақты бір есепке сәйкес келетін деректердің абстракцияларын табуға негізделеді. Осындай әрбір абстракциялар класс түрінде құрылады, класс -бағдарламалық жүйе құрылымының архитектуралық бірлігі болатын модуль» [5].

Windows қосымшалардың көпшілігінде кластар екі қызметті де орындайды, сондықтан бағдарламалық жүйенің әрбір модулінің өзінің нақты міндеті бар. C# тілінде деректер типі болатын және модуль қызметіндегі кластар қолданылады. Модуль кластарына, мысалы, Console, Math кластары жатады. Модуль ретіндегі кластар объекттерді құрай алмайды, ал нақтырақ айтқанда осындай кластың бір ғана объектісі болады. Осы модульдің өрістері мен әдістері басқа кластардың әдістеріне қолжетімді болады.

Үлкен бағдарламалық жүйелерді дайындағанда әзірлеу ортасы маңызды рөл атқарады. Бағдарламалау орталары ұсынатын бағдарламалау технологиялары күрделі бағдарламалық жүйелерді дайындау уақытын едәуір қысқартады.

Сондықтан ОББ бағдарламалау технологиясымен Visual Studio.NET бағдарламалау ортасында Windows (Windows Forms Application) қосымшаларын құра отырып танысатын боламыз.

1.2 Windows жүйесінің оқиғаларды басқару ұғымы

Консольді қосымшаларда бағдарлама жұмысы іске қосылғаннан кейін Main() әдісінің операторлары орындала бастайды. Windows үшін жазылған бағдарламардың ерекшелігі – бағдарлама іске қосылғаннан кейін ол Windows-тан келетін хабарларды күтудің шексіз цикліне ауысады.

Хабар дегеніміз - Windows операциялық жүйесінің жүйеде өтіп жатқан оқиғаларға жауабы. Оқиға ретінде компьютер жұмысында кез келген «бейстандарт» жағдайды есептеуге болады, мысалы, пернетақтада пернені басу, тышқан курсорының орнын ауыстыруы, нөлге бөлу, т.б.

Windows жүйесі жауап бере алатын барлық оқиғалар нөмірленген және әрбір нөмерге – «үзу векторына» сәйкес оқиғаға дұрыс жауап беретін арнайы бағдарлама сәйкестендірілген. Мысалы, компьютердің шалғай құрылғыларының драйверлері (пернетақта, тышқан, таймер).

Оқиға пайда болғанда Windows жүйесі оқиға «нөмерін» анықтайды және сәйкес драйверді іске қосады. Драйвер оқиғаны «өңдеп», Windows жүйесіне хабарлама жібереді.

Windows жүйесі жұмысының негізінде оқиғаларды басқару принципі жатыр. Сонымен, жүйе және Windows үшін жазылған барлық қосымшалар іске қосылғаннан кейін пайдаланушы іс-әрекеттерін немесе операциялық жүйенің оқиғаларын күтеді және оларға белгілі тәртіпте жауап қайтарады.

Windows хабары болған оқиға туралы жазба болып табылады. Мысалы, кейбір хабардың құрылымында мыналар болуы мүмкін: бағдарлама терезесінің дескрипторы, хабарлама коды (идентификаторы), анықтаушы параметрлер (мысалы, тышқан меңзерінің x пен y координаттары), хабарламаның құрылу уақыты.

Windows жүйесі қабылдайтын барлық хабарлар бір ғана данада болатын хабарлардың жүйелік кезегіне орналастырылады. Одан кейін жүйелік кезектен хабарлар жеке Windows қосымшаларының хабарлар кезегіне үйлестіріледі. Сонымен қатар, әрбір қосымша үшін өзінің хабарлар кезегі құрылады. Қосымшалардың хабарлар кезегі тек қана жүйелік хабарлардан толықтырылмайды. Кез келген қосымша хабарды кез келген басқа хабарға, сонымен қатар өзіне жібере алады. Әрбір Windows қосымшаның Windows-тан келетін, хабарларды өңдейтін үздіксіз циклі болады. Осы циклдің көмегімен қосымшалар «өзінің» хабарларын алады және қосымшаның тиісті хабарлар өңдеуішіне жібереді. Қосымшаның әрбір терезесінде хабарларды өңдейтін өз циклі және терезе функциясы (оған қосымша кезегінен алынатын хабарлар жіберіледі) болады.

Әдетте Windows қосымшасының негізгі терезесі болады, онда негізгі элементтер орналасады – меню, батырма, жалаушалар, т.б. Қосымшамен

жұмыс істеу барысында пайдаланушылар менюді таңдайды, батырмаларды басады немесе басқа басқару элементтерін қолданады.

Әрбір басқару элементінің өз идентификаторы болады. Мысалы, батырманы басқанда пайда болатын хабар Windows қосымшасының хабарлар кезегіне орналастырылады. Қолданылған басқару элементінен келетін хабарды Windows операциялық жүйесі осы басқару элементінің қосымшасының кезегіне жібереді.

Windows-та құрылатын қосымшаларда (File -> New -> Project -> Windows Forms Application) екі негізгі тип қолданылады: Form, Application.

Application класы қосымшаны басқарады: хабарларды өңдеу циклі (Application.Run();) бар Main() әдісін іске қосады, хабарды алғанда тиісті әрекеттерді орындайды және қосымша жұмысын дұрыс аяқтайды (Program.cs файлы).

Form класы пайдаланушы интерфейсін анықтайды: форма терезесін инициализациялайды, қосымшаны жұмысқа дайындайды (Form1.cs файлы).

Қарапайым Windows қосымшасын құру барысында әрекеттер ретін қарастырайық.

1.3 Visual Studio.Net ортасының негізгі терезелері

Шартты түрде Windows қосымшаларын құру процессі екі кезеңнен тұрады.

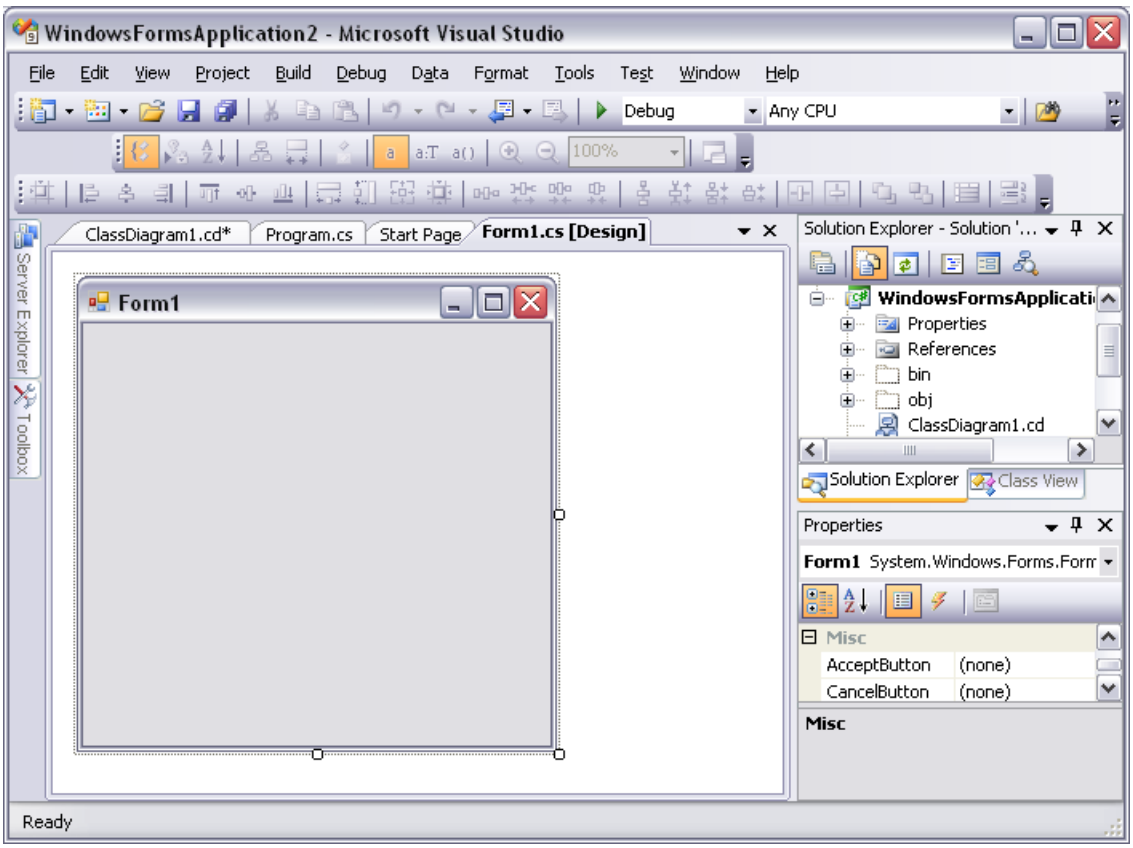
Бірінші кезең – пайдаланушы интерфейсіне арналған визуалды бағдарламалау кезеңі.

Екінші кезеңде қосымшаның хабарлар өңдеуішінің кодын құру керек, яғни Windows жіберетін хабарды алған кездегі қосымшаның жұмысын анықтау.

Windows қосымшаларын құру бойынша бірінші кезеңін орындау үшін Visual Studio .Net ортасын ашу керек (File -> New -> Project -> Windows Forms Application), 1.1-сурет. Жобаны дайындаған кезде жұмыс үстелінде бума атауын көрсету керек, онда барлық файлдар сақталады.

1.1-суретінің ортасында System.Windows.Forms атау кеңістігіне тиісті Form 1 терезесі орналасқан. Жобаны іске қосу үшін F5 пернесін басу керек немесе ортаның Debug жұмыс режімінің Start командасын таңдау керек. Іске қосылған қосымшаның терезесі 1.2-суретте көрсетілген.

Visual Studio .Net ортасының жеке терезелерінің қызметін толығырақ қарап шығайық.

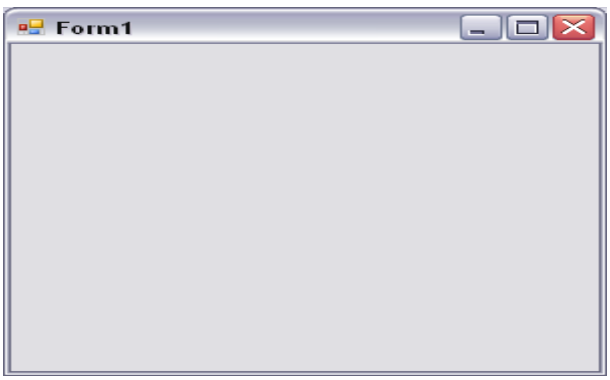


1.1-сурет – Visual Studio .Net ортасы

Form1 терезесі 4 беттен тұрады, олар жобаны редакциялаудың әртүрлі режімінде қолданылады, мысалы, Form1.cs[Design] элементтерді формада орналастыру үшін қолданылады, ал Program.cs терезесі бағдарлама кодын редакциялауда қолданылады.

Properties терезесінде басқару элементтерінің қасиеттері анықталады. Осы терезенің беттері категориялар, қасиеттер немесе оқиғалар бойынша топтастырылған.

Server Explorer терезесі (1.1-сурет, сол жақта, жиналған түрде көрсетілген) – сіздің компьютеріңіздегі, сервердегі деректер көздеріне қол жеткізу үшін қолданылады.



1.2-сурет – бағдарламаның жұмыс терезесі

Toolbox терезесінде (1.1-сурет, сол жақта, жиналған түрде) түрлі басқару элементтері болады.

Solution Explorer терезесі (1.1-сурет, оң жақта) жоба файлдарын қарауға және редакциялауға мүмкіндік береді. Онда жобаны файлдар және кластар бойынша қарауға болады.

Әдетте бағдарлама іске қосылғаннан кейін қателер терезесі пайда болады, онда бағдарлама кодында қате кеткен жолдың нөмірі шығады.

Visual Studio .Net ортасымен жұмыс жасаудың толық сипатттамасы А.В. Фролов, Г.В. Фролов «Визуальное проектирование приложений C#» кітабында берілген [3].

1.4 C# жобасын құру бойынша негізгі құрылымдық элементтер

C# тілі бағдарламалаудың объекті-бағытталған тіл деп аталады, онда класс негізгі ұғым болып табылады.

Класс дегеніміз – тип – көптеген объекттерді сипаттау үлгісі. Объект дегеніміз – класс типіндегі айнымалы, ол бағдарлама орындалу барысында динамикалық түрде құрылады, компьютер жадысында сақталады және бағдарлама аяқталғаннан кейін компьютер жадысынан өшіріледі.

Жобаны дайындаған кезде әдетте бірнеше класс дайындалады, бірақ жобаның жұмысы барысында бір-бірімен күрделі байланысқан жүздеген объекттер динамикалық түрде пайда болады.

FCL кітапханасының негізгі класы Object класы болып табылады, ол барлық кітапханалық және әзірлеуші дайындайтын кластардың түп тегі болып келеді.

Атаулар кеңістігі – бір тақырыптағы немесе әзірлеуші құрған белгілі бір кластар жиынтығының бірлесуі. Атаулар кеңістігіндегі кластардың атауы бірегей болуы керек. Әр түрлі атаулар кеңістігіндегі кластар атауы бірдей болуы мүмкін. Кластың толық атауы атаулар кеңістігінің атауынан, нүкте символынан және класс атауынан тұрады.

Атаулар кеңістігі FCL кітапханасының құрылымын жүйелейді. Егер барлық атаулар кеңістігін иерархиялық бұтақтар класы ретінде қарастырсақ, онда System кеңістігі, оның ішіндегі Object класы осы бұтақтың түпкі класы болады.

Жобаны құрастыру кезеңіндегі қосымша және компиляция бірлігі ретінде қарастыруға болады. Жоба компиляциясының нәтижесі құрастыру болады. Әрбір жобада бір немесе бірнеше атаулар кеңістіктері болады. Жобаны дайындайтын алғашқы кезеңінде белгіленген тип бойынша автоматты түрде қосымша қаңқасы құрылады, ол FCL кітапханасының құрамына кіретін кластардан тұрады. Егер "Windows Forms Application" типіндегі жоба құрылса, онда қосымша қаңқасына Form1 класы – кітапханалық Form класының мұрагері кіреді.

Жобаға автоматты түрде құрылған және жоба әзірлеушісі құрған барлық кластары бар файлдар кіреді. Сонымен қатар, жобада бағдарламаның

жұмысы барысында қолданылатын FCL кітапханасындағы атаулар кеңістігіне сілтеме болады. Жобада өзіне қосылатын барлық DLL-ге және басқа жобаларға сілтемелер болады.

«Жобаның жұмыс жасауы үшін оған керекті ресурстар мен орнатулар қосылады. Жобаны құрастыру сипаттамасын сақтайтын файл жобаның бір бөлігі болып келеді.

Таңдалған типіне байлысты жоба орындалатын және орындалмайтын болып бөлінеді. Орындалатын жобаларға, мысалы, Console немесе Windows типіндегі жобалар жатады. Орындалатын жоба қаңқасын құрғанда оған Main атаулы тұрақты әдісі бар класс қосылады. Осындай жобаны компиляциялау нәтижесінде PE-файл (Portable Executable file) – exe-ні анықтайтын, орындалатын файл құрылады. Ескерту, PE-файл компьютерде тек Framework .Net орнатылса ғана орындалады.» [5]

Орындалмайтын жобаларға, мысалы, Dll типіндегі жобалар жатады.

Құрастыру (сборка) – жоба компиляциясының нәтижесі. Ол нұсқаның нөмерімен белгіленген бір немесе бірнеше файлдар коллекциясынан тұрады. Әрбір құрастыру компьютерде біртұтас болады. Бағдарламашы жобамен жұмыс істесе, ал CLR құрастырулармен жұмыс істейді. Құрастыру қауіпсіздік сұрақтарын шешеді, өйткені онда өзіне керекті ресурстардың сипаттамасы мен элементтерді пайдалану құқықтары бар. Әрбір құрастыруда манифест болады, манифест құрастырудан және оның элементтерінің толық сипаттамасынан, қажетті ресурстардан, басқа құрастыруларға сілтемелерден тұрады. CLR-дің осы сипаттамасының арқасында құрастыруды өрбіту, аралық код трансляциясы мен оның орындалуы үшін басқа қосымша ақпарат қажет емес. Манифест құрастыруды идентификациялайды, құрастыруды орындау үшін керекті файлдарды спецификациялайды, құрастыруды құрайтын типтер мен ресурстарды спецификациялайды.

Visual Studio.NET 2008 ортасында дайындалатын әрбір жоба Шешім – Solution деп аталатын белгілі бір қоршамға орналастырылады. Шешімде, әдетте, ортақ тақырып бойыша байланыстырылған бірнеше жоба болуы мүмкін. Мысалы, бір Шешімге үш жобаны орналастыруға болады: әзірленген кластары бар DLL, консольді жоба, Windows-та басқарылатын жоба.

Жаңа жоба құрылу барысында оны бар Шешімге орналастыруға болады немесе ол үшін жаңа Шешім құрылады. [5]

1.5 Бірінші бағдарламаны құру мысалы

Windows жүйесінде жұмыс істейтін қосымшаның формасын қолдану мысалын қарастырайық. Мысал ретінде үшбұрыш периметрін есептеу есебі қарастырылады.

1.1-есеп. диалог режімінде үшбұрыш қабырғаларын беру және оның периметрін есептеу керек. Үшбұрыш қабырғаларын берілгеннен кейін мына тексерістерді орындау керек: үшбұрыш қабырғалары нөлден үлкен болуы керек және үшбұрыштың кез келген екі қабырғаларының қосындысы үшінші

қабырғадан үлкен болуы керек. Қосымша кодына түсініктемелерді қолдану керек.

Біз Toolbox терезесінен стандартты үш басқару элементін пайдаланамыз: тұрақты мәтін (Labe1) элементі, мәтінді енгізу-шығару өрісі (TextBox**)**, командалық батырма (Button).

– тұрақты мәтін;

– мәтінді енгізу-шығару өрісі;

– командалық батырма.

Түсіндіретін сөздерді жазу үшін төрт тұрақты мәтін қолданылады.

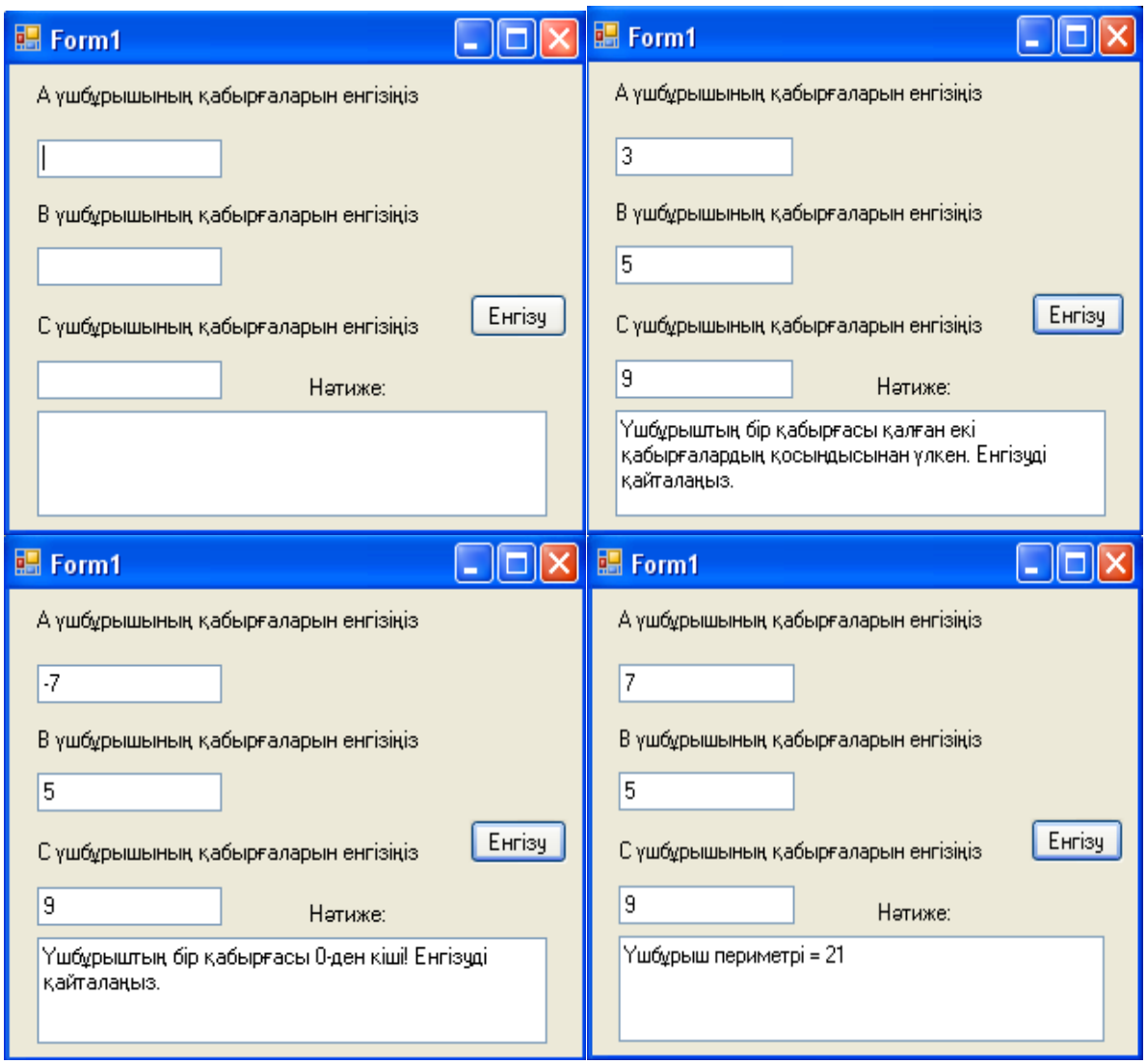
Үш мәтінді енгізу, нәтижені шығарудың бір өрісі және бір командалық батырма қолданылады.

Визуалды бағдарламалау процессінде формаға Toolbox терезесінен керекті басқару элементі көшіріледі және белгілі бір орынға орналастырылады. Әдетте Toolbox терезесі «жиналған» күйде болады. Оны «ашу» үшін тышқанның оң жақ пернесімен Toolbox панелін басу керек, элементі (оны басу керек) арқылы экранның белгілі бір орнына орнықтыруға болады. Жұмыс аяқталғаннан кейін элементінің көмегімен Toolbox терезесін «жинауға» болады.

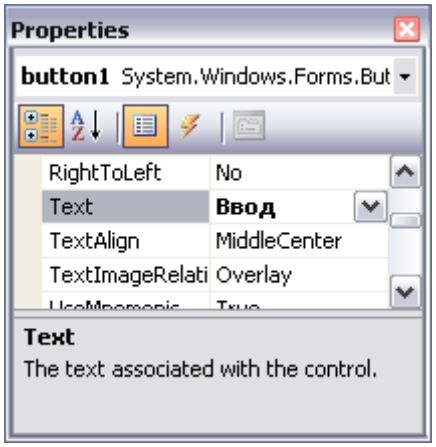
Визуалды бағдарламалау процесінде басқару элементтерінің кейбір қасиеттерін өзгерту керек, мысалы, тұрақты мәтін мен батырманың Text қасиеті өзгертілді (1.3-сурет). Ол үшін Properties терезесін пайдалану керек (1.4-сурет).

Енгізу өрістері мен шығару өрісінің айырмашылығы бар, шығару өрісінің `Multiline` қасиеті `Multiline = true` тең. Барлық басқару элементтерінде `Text` қасиеті қолданылды.

«Ввод» Батырмасын басу бойынша хабарды өңдеуші әдісін құру үшін визуалды бағдарламалау кезеңінде осы батырманы екі рет басса жеткілікті. Бос `private void button1_Click(object sender, EventArgs e)` хабар өңдеушісіне кодты жазамыз: үшбұрыш қабырғаларын мәндерін диалог режімінде беру және олардың үшбұрыш шарттарына сай келуі.



1.3-сурет – «Үшбұрыш» қосымшасының терезесі



1.4-сурет – button1 элементінің Properties терезесі

Program.cs файлының коды:

```
using System;
using System.Collections.Generic;
```

```
using System.Linq;
using System.Windows.Forms;
namespace WindowsFormsApplication1
{
 static class Program
 {
 /// <summary>
 /// The main entry point for the application.
 /// </summary>
 [STAThread]
 static void Main()
 {
 Application.EnableVisualStyles();
 Application.SetCompatibleTextRenderingDefault(false);
 Application.Run(new Form1());
 }
 }
}
```

Form1.cs файлының коды:

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Text;
using System.Windows.Forms;
namespace WindowsFormsApplication1 //Zadacha1_1
{
 public partial class Form1 : Form
 {
 public Form1()
 {
  InitializeComponent();
 }
 private void button1_Click(object sender, EventArgs e)
 {
  int a, b, c, p;
  a = Convert.ToInt32(textBox1.Text);
  b = Convert.ToInt32(textBox2.Text);
  c = Convert.ToInt32(textBox3.Text);
  p = a + b + c;
  if (a > 0 && b > 0 && c > 0)
  if (a + b > c && a + c > b && b + c > a)
   textBox4.Text = "Үшбұрыш периметрі = " + p.ToString();
  else
  {
   textBox4.Text = "Үшбұрыштың бір қабырғасы қалған екі
қабырғалардың қосындысынан үлкен. Енгізуді қайталаңыз.";
  }
  else
  {
```

```
        textBox4.Text = "Үшбұрыштың бір қабырғасы 0-ден кіші!
Енгізуді қайталаңыз.";
        }
       }
       private void Form1_Load(object sender, EventArgs e)
       {
       }
       }
      }
```

Орта автоматты түрде атаулар кеңістігін құрады. Оларды толығырақ қарастырайық.

System атаулар кеңісті базалық және іргелі анықтамалардан тұрады, олар: деректер типі, оқиғалар, оқиғалар өңдеуіштері, т.б.

System.Collections атаулар кеңістігінде кластар анықталған, олар массивтерді, тізімдерді, сөздіктерді, хэштерді анықтайтын контейнерлер қызметін атқарады.

System.ComponentModel кеңістігіндегі кластар қосымшаның компоненттері мен басқару элементтерінің белгілі бір тәртіптегі қызметін орындау үшін қолданылады.

System.Data класы ADO.NET интерфейсі арқылы деректер базасымен жұмыс істейтін қосымшалар үшін керек.

System**.**Drawing кеңістігі графикалық құрылғылар интерфейсіне (Graphics Device Interface, GDI) қол жеткізу үшін керек, нақтырақ айтсақ, оның толықтырылған GDI+ версиясы үшін. Осы кеңістіктегі кластар қосымша терезесінде сызықтарды, екі өлшемді пішіндерді, кескіндерді, басқа да графикалық объекттерді салу үшін керек.

System.Windows.Forms кеңістігінде формалардың жұмыс тәртібін орындайтын кластар анықталған.

Қосымшаға нақтысында System және System.Windows.Forms екі кеңістігі қажет, ал қалған атаулар кеңістігі қосымшаға қажетінше қосылады.

Қосымша ретке келтірілгеннен (после отладки) кейін барлық файлдарды сақтау керек (менюде File->Save All командасын таңдау керек).

Бағдарламалаудың визуалды ортасы кішігірім қосымша үшін де 10-нан аса файлдар мен бумаларды дайындайды. Жұмыс үстеліндегі бірінші бағдарламаның 1_1_treygolnik бумасында WindowsFormsAplication1 бумасы бар. Оның ішінде WindowsFormsAplication1 атты бума мен WindowsFormsAplication1.csproj-ды редакциялау үшін шақырылатын жоба файлы бар. Ал WindowsFormsAplication1 бумасында тағы bin, obj, Properties бумалары мен бірнеше файлдар (бағдарлама коды – Program.cs, форма коды – Form1.cs) бар. Осы бумада Form1 формасы бойынша файлдың сырт пішінін сақтайтын ресурстық файл және форма мен онда орналасқан барлық басқару элементтер «қасиеттерінің» мәндерін сақтайтын Form1.Designer.cs файлы орналасады.

Visual Studio .Net ортасымен жұмыс жасау үшін әзірше бізге тек форма кодының файлы – Form1.cs қажет және осы файлды ғана өзгертуге болады, басқа файлдардың атауы мен қасиеттерін берілген күйде қалдыру керек.

1.6 Өзін-өзі тексеру сұрақтары

1 Компьютер жұмысындағы оқиғалар ұғымы?

2 Windows жүйесі оқиғаларды қалай «ажыратады»?

3 Windows жүйесі оқиғаның пайда болуы туралы ақпаратты алғанннан кейін не істейді?

4 Драйвер ұғымы.

5 Хабар ұғымы .

6 Хабар неден тұрады?

7 Windows жүйесі драйверден қабылдап алатын хабарларды қайда жібереді?

8 Әрбір қосымшада Windows-тан келетін хабарларды өңдеу циклі не үшін қолданылады?

9 Қосымшаның қандай әдісі Windows-тан келетін хабарларды өңдеу циклін жүзеге асырады?

10 Form класы қандай мақсатпен қолданылады?