8 дәріс. Мұралау. Мұралау және атауларды жасыру. Көпдеңгейлі иерархияны құру.

Дәрістің мақсаты: студенттерде кластар иерархиясын құру мақсатында мұралау механизмінің идеясы туралы түсініктерін көрсетуге қабілет қалыптастыру.

Осы дәрісті меңгеру нәтижесінде студенттер келесі қабілеттерге ие болады:

- Мұралау механизмінің идеясы бойынша түсініктерін көрсету;
- Көпдеңгейлі иерархияны құру ерекшеліктері бойынша түсініктерін көрсету;
- Мұралау кезінде атауларды жасыру механизмі бойынша түсініктерін көрсету.

Мұралау – объектіге бағытталған программалаудың негізгі қағидаларының бірі, ол иерархиялық жіктеуді құруға мүмкіндік береді. Мұралаудың арқасында байланысқан элементтер жиынына тән ерекшеліктер анықталатын жалпы класты құруға болады. Бұл кластан кейіннен өздерінің ерекшеліктерін қосу арқылы басқа, айтарлықтай нақтыланған кластар мұралай алады.

C# тілінде мұраланатын класс базалық, ал мұралайтын класс туынды класс деп аталады. Ендеше, туынды класс базалық кластың арнайы нұсқасы болып табылады. Ол базалық класта анықталатын барлық айнымалылар, әдістер, қасиеттер және индексаторларды мұралайды да, оларға өзіндік элементтерін қосады.

C# тілінде мұралауды қолдау бір кластың жариялануына екінші класты енгізу арқылы жүзеге асырылады. Ол үшін туынды класты жариялау кезінде базалық класс көрсетіледі. Төменде базалық кластан мұралайтын туынды класты жариялаудың жалпы формасы берілген:

```
class туынды_класс_аты : базалық_класс_аты  {
   // класс тұлғасы
}
```

Туынды кластың жариялануындағы қоснүкте : белгісінің қолданылуына назар аударыңыз. Кез келген туынды класс үшін бір ғана базалық класты көрсетуге болады. Алайда мұралау иерархиясын құрып, туынды класты басқа класс үшін базалық класс ретінде пайдалану мүмкіндігі бар.

1-мысал. Өнімнің атын, өндірушісін, бағасын сипаттайтын базалық класс берілген. Осы ақпаратқа қосымша өнімнің дүкендегі бар саны берілетін Тауар класын құру керек делік.

```
using System;
// Өнімдерді сипаттауға арналған класс
class Onim {
   public string aty;
   public string ondirushi;
   public double bagasy;
   public void Shygaru()
   {
      Console.Write(ondirushi + " ondirushisining " + aty + "
                    onimining bagasy – " + bagasy);
   }
}
// Өнім класынан туынды тауар класы
class Tauar : Onim {
   public int sany;
   // тауар туралы ақпаратты шығару
   new public void Shygaru() {
      base.Shygaru(); Console.WriteLine(", al sany – " + sany);
   }
   // Берілген тауардың жалпы бағасы
   public void zhalpy_baga()
```

```
        {
            Console.WriteLine("Tauardyng zhalpy bagasy: " +
                              bagasy * sany);
        }
    }
    class Program {
        static void Main() {
            Onim onim001 = new Onim();
            onim001.aty = "notebook";
            onim001.ondirushi = "Acer";
            onim001.bagasy = 200000;
            onim001.Shygaru(); Console.WriteLine();
            Tauar tauar001 = new Tauar();
            tauar001.aty = "notebook";
            tauar001.ondirushi = "Acer";
            tauar001.bagasy = 200000;
            tauar001.sany = 15;
            tauar001.Shygaru();
            tauar001.zhalpy_baga();
            Console.ReadKey();
        }
    }
```

Бұл программаның орындалу нәтижесі келесідей болады:

```
    Acer ondirushisining notebook onimining bagasy – 200000
    Acer ondirushisining notebook onimining bagasy – 200000, al
sany – 15
    Tauardyng zhalpy bagasy: 3000000
```

Program класында **Onim** және **Tauar** екі класының объектілері құрылады. Мұндағы **Tauar** класының құрамында тауардың аты, өндірушісі және бағасы туралы айнымалылар анықталмағанына қарамастан, класс объектісін құрғаннан кейін ол үшін аталған параметрлердің мәнін көрсетуге болады, бұл экземпляр айнымалылары базалық кластан мұраланады. Базалық кластың әдістерін де туынды класта қайта жарияламастан пайдалануға болады. Жоғарыдағы мысалда базалық кластың **Shygaru()** әдісі туынды класта толықтыруды қажет ететіндіктен, ол тікелей мұраланбай, қайта анықталады. Әдісті қайта анықтау арқылы оның қызметін туынды класта өзгертуге болады. Базалық кластың әдістері туынды класта **new** кілттік сөзінің көмегімен қайта анықталады, қайта анықталған әдістің құрамында базалық кластың әдісін шақыру үшін **base** кілттік сөзі қолданылады. Туынды класта **Shygaru()** әдісінен басқа, берілген тауардың дүкенде бар саны үшін жалпы бағаны анықтайтын **zhalpy_baga()** әдісі анықталған, базалық класс құрамына кірмейтін әдіс туынды класта әдеттегідей түрде анықталады.

Мысалда көрсетілгендей, **Onim** класы **Tauar** класы үшін базалық класс болғанына қарамастан, ол өз алдына тәуелсіз класс болып табылады және әдеттегі класс сияқты қолданыла алады. Бірақ **Onim** класының объектілері оның туынды кластарымен байланыспайды.

Мұралауда кластың мүшелеріне қатынасу

Өткен тарауларда көрсетілгендей, класс мүшелерін рұқсатсыз қолданудың алдын алу үшін оларды көп жағдайда жабық түрде жариялайды. Класты мұралау бұл шектеулерден тыс қалмайды, туынды класс құрамында базалық кластың жабық мүшелері қол жетімсіз болады.

Аталған шектеуді жеңудің жолдары – қорғалған (**protected**) класс мүшелерін пайдалану немесе жабық мәліметтерге қол жеткізу үшін ашық қасиеттерді қолдану. Базалық класта жарияланған әдістер сияқты, ашық қасиеттер де туынды кластарда

қолжетімді болады. Базалық класс айнымалыларына ашық қасиеттердің көмегімен қатынасудың кемшілігі – туынды класта бұл айнымалыларды тікелей пайдалана алмаймыз.

2-мысал. Базалық кластың жабық мүшелерінің мәнін тағайындау және шығарып алу үшін ашық қасиеттерді пайдалану

```
using System;
// Өнімдерді сипаттауға арналған класс
class Onim {
   private string aty;
   private string ondirushi;
   private double bagasy;
   // Қасиеттер
   public string Aty
   {
      get { return aty;}
      set {aty = value;}
   }
   public string Ondirushi
   {
      get { return ondirushi;}
      set { ondirushi = value;}
   }
   public double Bagasy
   {
      get { return bagasy;}
      set { if(value>0) bagasy = value;
            else bagasy = 0;}
   }
   public void Shygaru()
   {
      Console.Write(ondirushi + " ondirushisining " + aty + "
                  onimining bagasy – " + bagasy);
   }
}
// Өнім класынан туынды тауар класы
class Tauar : Onim {
   public int sany;
   // тауар туралы ақпаратты шығару
   new public void Shygaru()
   {
      base.Shygaru(); Console.WriteLine(", al sany – " + sany);
   }
   // Берілген тауардың жалпы бағасы
   public void zhalpy_baga()
   {
      Console.WriteLine("Tauardyng zhalpy bagasy: "
                        + Bagasy * sany);
   }
}
class Program {
   static void Main() {
      Onim onim001 = new Onim();
      onim001.Aty = "notebook";
      onim001.Ondirushi = "Acer";
      onim001.Bagasy = 200000;
      onim001.Shygaru(); Console.WriteLine();
      Tauar tauar001 = new Tauar();
      tauar001.Aty = "notebook";
      tauar001.Ondirushi = "Acer";
      tauar001.Bagasy = 200000;
      tauar001.sany = 15;
      tauar001.Shygaru();
      tauar001.zhalpy_baga();
```

```
        Console.ReadKey();
    }
}
```

Бұл программаның орындалу нәтижесі келесідей болады:

```
    Acer ondirushisining notebook onimining bagasy – 200000
    Acer ondirushisining notebook onimining bagasy – 200000,
al sany – 15
    Tauardyng zhalpy bagasy: 3000000
```

Көрсетілген мысалда **Aty**, **Ondirushi**, **Bagasy** ашық қасиеттері кластың **aty**, **ondirushi**, **bagasy** жабық мүшелеріне қатынасуға мүмкіндік береді.

Базалық класта жарияланған жабық мүшелерді туынды класта пайдаланудың келесі тәсілі қорғалған мүшелерді пайдалану екендігі жоғарыда айтылған. Кластың қорғалған мүшесі кластар иерархиясы аясында ашық, ал иерархиядан тыс жабық болады. Кластың қорғалған мүшесі **protected** қатынасу модификаторының көмегімен құрылады.

3-мысал. **protected** қатынасу модификаторын пайдалану мысалы.

```
using System;
// Өнімдерді сипаттауға арналған класс
class Onim {
   protected string aty;
   protected string ondirushi;
   protected double bagasy;
   public void manderi(string a, string o, double b)
   {
      aty = a, ondirushi = o, magasy = b;
   }
   public void Shygaru()
   {
      Console.Write(ondirushi + " ondirushisining " + aty + "
                    onimining bagasy – " + bagasy);
   }
}
// Өнім класынан туынды тауар класы
class Tauar : Onim {
   private int sany;
   public int Sany
   {
      get { return sany;}
      set { if(value > 0) sany = value;
            else sany=0; }
   }
   // тауар туралы ақпаратты шығару
   new public void Shygaru()
   {
      base.Shygaru(); Console.WriteLine(", al sany – " + sany);
   }
   // Берілген тауардың жалпы бағасы
   public void zhalpy_baga()
   {
      Console.WriteLine("Tauardyng zhalpy bagasy: "
                        + bagasy * sany);
   }
}
class Program {
   static void Main() {
```

```
        Onim onim001 = new Onim();
        onim001.manderi("notebook", "Acer", 200000);
        onim001.Shygaru(); Console.WriteLine();
        Tauar tauar001 = new Tauar();
        tauar001.manderi("notebook", "Acer", 200000);
        tauar001.Sany = 15;
        tauar001.Shygaru();
        tauar001.zhalpy_baga();
        Console.ReadKey();
    }
}
```

Бұл программаның орындалу нәтижесі келесідей болады:

```
    Acer ondirushisining notebook onimining bagasy – 200000
    Acer ondirushisining notebook onimining bagasy – 200000,
al sany – 15
    Tauardyng zhalpy bagasy: 3000000
```

Protected қатынасу модификаторын тек кластар иерархиясына қолжетімді, ал кодтың басқа бөліктері үшін қолжетімсіз класс мүшелерін құру үшін ғана пайдаланған дұрыс. Егер кодтың кластар иерархиясынан тыс бөлігінде жабық мүшелердің мәнін тағайындау мүмкіндігін қарастыру қажет болса, ондай мүшелерге қатынасуды қасиеттердің көмегімен ұйымдастырған жөн.

Мұралаудағы конструкторлар

Кластар иерархиясында базалық және туынды кластардың өзіндік конструкторларын құру мүмкіндігі бар. Туынды класс үшін жеке конструктор құрған жағдайда базалық класс конструкторы объектінің базалық бөлігін, ал туынды класс конструкторы осы объектінің туынды бөлігін қалыптастыратын болады.

Базалық және туынды кластың екеуінде де конструкторлар анықталған жағдайда, туынды класс конструкторы **base** кілттік сөзінің көмегімен құрылады. Туынды класс конструкторын жариялаудың кеңейтілген формасын және **base** кілттік сөзін пайдалану арқылы туынды класта базалық класс конструкторын шақыруға болады. Конструкторды жариялаудың кеңейтілген формасы төменде көрсетілген:

```
    туынды_класс_конструкторы(параметрлер_тізімі) : base
(аргументтер_тізімі)
    {
        // конструктор тұлғасы
    }
```

мұндағы *аргументтер_тізімі* базалық класс конструкторына қажетті кез келген аргументтерді білдіреді. Бұл мүмкіндікті пайдалану ерекшеліктерін мысал арқылы қарастырайық.

4-мысал. Өнім және тауар кластар иерархиясында конструкторларды пайдалану

```
using System;
// Өнімдерді сипаттауға арналған класс
class Onim {
   private string aty;
   private string ondirushi;
   private double bagasy;
   public string Aty
   {
      get { return aty;}
      set {aty = value;}
```

```
    }
    public string Ondirushi
    {
       get { return ondirushi;}
       set { ondirushi = value;}
    }
    public double Bagasy
    {
       get { return bagasy;}
       set { if(value > 0) bagasy = value;
             else bagasy = 0;}
    }
    public Onim(string a, string o, double b)
    {
       aty = a; ondirushi = o, bagasy = b;
    }
    public void Shygaru()
    {
       Console.Write(ondirushi + " ondirushisining " + aty + "
                     onimining bagasy – " + bagasy);
    }
}
// Өнім класынан туынды тауар класы
class Tauar : Onim {
    private int sany;
    public int Sany
    {
       get { return sany;}
       set { if(value > 0) sany = value;
             else sany = 0;}
    }
    public Tauar(string a, string o, double b,
                                    int s) : base(a, o, b)
    {
       sany = s;
    }
    // тауар туралы ақпаратты шығару
    new public void Shygaru()
    {
       base.Shygaru(); Console.WriteLine(", al sany – " + sany);
    }
    // Берілген тауардың жалпы бағасы
    public void zhalpy_baga()
    {
       Console.WriteLine("Tauardyng zhalpy bagasy: "
                         + Bagasy * sany);
    }
}
class Program {
    static void Main() {
       Onim onim001 = new Onim("notebook", "Acer", 200000);
       onim001.Shygaru(); Console.WriteLine();
       Tauar tauar001 = new Tauar("notebook", "Acer", 200000, 15);
       tauar001.Shygaru();
       tauar001.zhalpy_baga();
       Console.ReadKey();
    }
}
```

Бұл программаның орындалу нәтижесі төмендегідей болады:

```
    Acer ondirushisining notebook onimining bagasy – 200000
    Acer ondirushisining notebook onimining bagasy – 200000,
al sany – 15
```

```
Tauardyng zhalpy bagasy: 3000000
```

Көрсетілген мысалда **Tauar()** конструкторы **a**, **o**, **b** параметрлері бар **base** әдісін шақырады. Бұл әрекет **Aty**, **Ondirushi**, **Bagasy** қасиеттерін **a**, **o**, **b** параметрлерінің мәндерімен инициалдайтын **Onim()** конструкторының шақырылуына алып келеді.

base кілттік сөзінің көмегімен базалық класта анықталған кез келген конструктор түрін шақыруға болады, бұл жағдайда берілген аргументтерге сәйкес келетін параметрлері бар конструктор шақырылады.

Сонымен, конструкторларды төмендегідей анықтауға болады:

Базалық класс конструкторы	***Туынды класс конструкторы***
`public Onim() { aty = " ", ondirushi = " ", bagasy = 0;}`	`public Tauar() { sany = 0;} /* автоматты түрде келісім бойынша конструктор шақырылады */`
`public Onim(string a) { aty = a, ondirushi = " ", bagasy = 0;}`	`public Tauar(string a, int s): base(a) { sany = s;}`
`public Onim(string a) { aty = a, ondirushi = " ", bagasy = 0;}`	`public Tauar(string a): base(a) { sany = 0;}`
`public Onim(string a, string o, double b) { aty = a, ondirushi = o, bagasy = b;}`	`public Tauar(string a, string o, double b, int s): base(a, o, b) { sany = s;}`

Base кілттік сөзі көпдеңгейлі иерархияда шақырушы кластан бір саты жоғары тұрған базалық класс конструкторын шақырады. Егер **base** кілттік сөзі көрсетілмесе, базалық кластың келісім бойынша конструкторы шақырылады.