Nesne Tabanlı Programlama
Bölüm 11
C# Indexer

Dr. Öğr. Üyesi Murat TAŞYÜREK (kayubmprogramlama1@gmail.com)

18 Aralık 2023

Kayseri Üniversitesi, Bilgisayar Mühendisliği Bölümü

Ders İçeriği

- Indexer
- Generic Indexer
- Overload Indexer

Indexer

- **Indexer** bir sınıfa dizi gibi erişilmesine olanak tanıyan bir özellik türüdür.
- **Indexer** özel tanımlı bir property olarak da ifade edilir.
- Sadece class içerisinde tanımlanabilir.
- Tanımlandığı sınıfı indexlenebilir özelliği getirir.
- Array işlemlerinde kullandığımız [] operatörünü sınıf için kullanmamıza olanak sağlar.
- **Index** integer bir değer de olabilir. String bir değer de olabilir.
- Sınıf içersinde bir nevi dinamik alan tanımlarsınız. Geniş kapsamlı projelerde bu nedenle yaygın kullanılır.

Indexer Kullanımı

```
0 references
public void DbGetTableValue()
{
    SqlConnection con = new SqlConnection("Connection string");
    con.Open();
    SqlCommand cmd = new SqlCommand("Select * from Table", con);
    SqlDataAdapter da = new SqlDataAdapter(cmd);
    DataSet ds = new DataSet();
    da.Fill(ds);
    string value = "";
    foreach (DataRow r in ds.Tables[0].Rows)
    {
        value = r["DbColumName"].ToString();
    }

    con.Close();

}
```

Indexer kullanımı

- C# programlama dilinde **Indexer** aşağıda gösterildiği gibi kullanılır.

```
[access_modifier] [return_type] this[parameter_type parameter_name]
{
   get
   {
     // return specified index value
   }
   set
   {
     // set a value at the specified index
   }
}
```

Indexer kullanımı

- **access modifier:** public, protected veya private olabilir.
- **return type:** C# dilinde tarafından desteklenen her hangi bir tür olabilir.
- **this:** Geçerli sınıfın nesnesine işaret eden anahtar kelimedir.
- **argument list:** Bu, indeksleyicinin parametre listesini belirtir. Integer veya string olabilir.
- **get**{ } **and set** { }: Bu erişimciler sayesinde değer atanır veya okunur.

Örnek

- En fazla 5 adet string değer tutabilen bir indexer içeren sınıfı tanımlayalım.
- Tanımlı olan dizi private olsun.
- Indexer public olsun ve integer değer alsın.
- Kodlayalım

Class

```
class Ornek2Class
{
    private string[] strArr = new string[5]; // internal data storage, encapsulation

    0 references
    public string this[int index]
    {
        get
        {
            if (index < 0 || index >= strArr.Length)
                throw new IndexOutOfRangeException("Index out of range");

            return strArr[index];
        }

        set
        {
            if (index < 0 || index >= strArr.Length)
                throw new IndexOutOfRangeException("Index out of range");

            strArr[index] = value;
        }
    }
}
```

Source code

```
namespace NNDers10Ornek2
{
    0 references
    internal class Program
    {
        0 references
        static void Main(string[] args)
        {
            Ornek2Class strStore = new Ornek2Class();

            strStore[0] = "One";
            strStore[1] = "Two";
            strStore[2] = "Three";
            strStore[3] = "Four";
            strStore[3] = "Five";

            for (int i = 0; i < 5; i++)
                Console.WriteLine(strStore[i]);
        }
    }
    2 references
    class Ornek2Class...
}
```

Çıktı

```
0 references
internal class Program
{
    0 references
    static void Main(string[] args)
    {
        Ornek2Class strStore = new Ornek2Class();

        strStore[0] = "One";
        strStore[1] = "Two";
        strStore[2] = "Three";
        strStore[3] = "Four";
        strStore[4] = "Five";
    }
}
2 references
class Or
```

```
C:\WINDOWS\system32\cmd.exe
One
Two
Three
Four
Five
Press any key to continue . . .
```

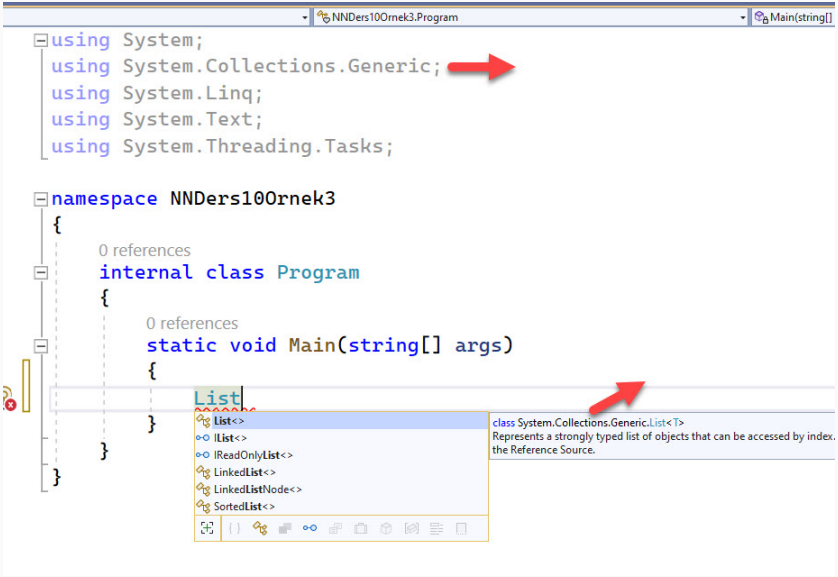
C# Generic

- **Generics** tasarlandığımız interface, class, metod yada parametrelerin belirli bir tip için değil bir şablon yapısına uyan her tip için çalışmasını sağlayan bir yapıdır
- Kaliteli ve daha yönetilebilir kod yazmamıza olanak sağlar.
- Çalışma zamanında (run time) gereksiz Cast ve Boxing-Unboxing kullanmasını önlediğinden efektif performans sağlar.
- Derleme zamanında (compile time) (type safe) tip güvenli değişken kullanılmasını zorlayarak çalışma zamanında oluşabilecek tip dönüşüm hatalarını önler.
- Programcıya kod üzerinde daha güçlü esnek bir kontrol sağlar.

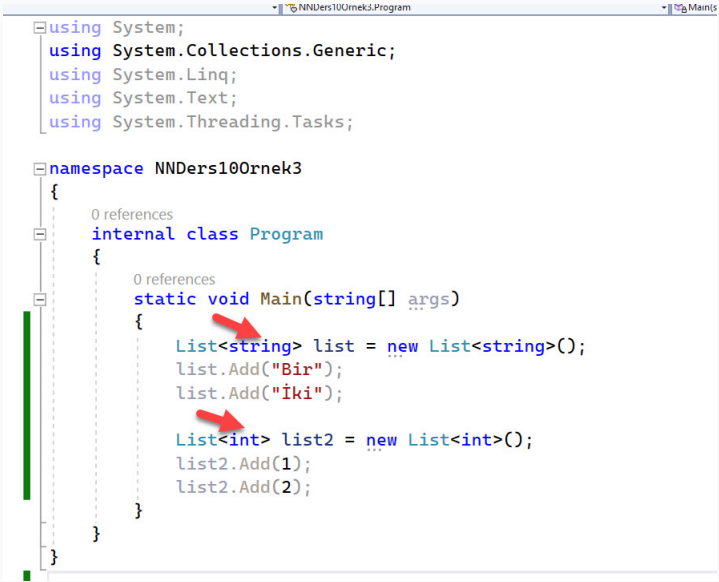
C# Generic

- Generic means the general form, not specific. In C#, generic means not specific to a particular data type.
- Generic <T> ile tanımlanır.
- Bu şekilde tanımlanan bir sınıfa daha sonra istediğiniz türde değer gönderebilirsiniz. Ancak her nesne için sadece bir tür kullanmanız gerekir.
- Sadece tek tür veri tuttuğundan dolayı daha verimli çalışmaktadır.
- Eğer koleksiyon içinde bütün veriler aynı tipte olacaksa ArrayList yerine Generic List koleksiyon yapısı kullanılır.
- C#'da List<T> sınıfı gereric olarak tanımlandığından istediğiniz türü ekleyebilmektesiniz.

Generic List



Örnek Generic List



Örnek

- Bir sınıf tanımlayalım.
- Tanımladığımız sınıfın Indexer'ı olsun.
- Ancak bu sınıfımız Gereric olsun ve istediğimiz değerleri tutalım.
- Tutacağımız değer sayısı da nesne oluşturulurken belirlenebilsin. Eğer belirli değilse default 7 olsun.
- Kodlayalım.

Class

```
2 references
class DataStore<T>
{
    private T[] store;
    0 references
    public DataStore()
    {
        store = new T[7];
    }
    0 references
    public DataStore(int length)
    {
        store = new T[length];
    }
    0 references
    public T this[int index]
    {
        get
        {
            if (index < 0 && index >= store.Length)
                throw new IndexOutOfRangeException("Index out of range");

            return store[index];
        }
        set
        {
            if (index < 0 || index >= store.Length)
                throw new IndexOutOfRangeException("Index out of range");

            store[index] = value;
        }
    }
    0 references
    public int Length
    {
        get
        {
            return store.Length;
        }
    }
}
```

Source code

```
6
7   namespace NNDers10Ornek4
8   {
        0 references
9       internal class Program
10      {
            0 references
11          static void Main(string[] args)
12          {
13              DataStore<int> grades = new DataStore<int>();
14              grades[0] = 100;
15              grades[1] = 25;
16              grades[2] = 34;
17              grades[3] = 42;
18              grades[4] = 12;
19              grades[5] = 18;
20              grades[6] = 2;
21
22
23              for (int i = 0; i < grades.Length; i++)
24                  Console.WriteLine(grades[i]);
25
26              DataStore<string> names = new DataStore<string>(5);
27              names[0] = "Ahmet";
28              names[1] = "Mehmet";
29              names[2] = "Murat";
30              names[3] = "Efe";
31              names[4] = "Yiğit";
32
33              for (int i = 0; i < names.Length; i++)
34                  Console.WriteLine(names[i]);
35          }
36      }
        6 references
37      class DataStore<T>...
73  }
```

Çıktı

```
4                                                    NNDers10Ornek4.Program
using System.Text;
using System.Threading.Tasks;

namespace NNDers10Ornek4
{
    0 references
    internal class Program
    {
        0 references
        static void Main(string[] args)
        {
            DataStore<int> grades = ne
            grades[0] = 100;
            grades[1] = 25;
            grades[2] = 34;
            grades[3] = 42;
            grades[4] = 12;
            grades[5] = 18;
            grades[6] = 2;


            for (int i = 0; i < grades
                Console.WriteLine(grad

            DataStore<string> names =
            names[0] = "Ahmet";
            names[1] = "Mehmet";
```

```
C:\WINDOWS\system32\cmd.exe
100
25
34
42
12
18
2
Ahmet
Mehmet
Murat
Efe
Yiğit
Press any key to continue . . .
```

Overload Indexer

- Farklı tipte girdi verisi alarak fonksiyonların overload edilebildiği gibi Indexer'da overload edilebilir.
- Örneğin veritabanı bağlantı katmanı için bir sınıf yazalım. Sınıfımız Rows dizisinde RowResult türünde nesneleri tutabilsin.
- Ancak bu nesnelere sadece Indexer ile erişebilsin.
- Indexer ile de ID veya KeyWord üzerinden erişebilsin.
- RowResult sınıfınında ID (int) , KeyWork (string) ve Value (object) özellikleri olsun.
- Kodlayalım

Class

```
11 references
public class RowResult
{
    public int ID;
    public string Keyword;
    public object Value;
}
4 references
public class DbResult
{
    private RowResult[] Rows;

    0 references
    public DbResult()
    {
        Rows = new RowResult[2];
    }
    1 reference
    public DbResult(int lenght)
    {
        Rows = new RowResult[lenght];
    }
    3 references
    public int RowCount
    {
        get => Rows.Length;
    }
    4 references
    public RowResult this[int index]...
    2 references
    public RowResult this[string KeyWord]...
}
```

Class Int Indexer

```
1 reference
public DbResult(int lenght)
{
    Rows = new RowResult[lenght];
}
3 references
public int RowCount
{
    get => Rows.Length;
}
4 references
public RowResult this[int index]
{
    get
    {
        if (index < 0 || index >= RowCount)
            throw new Exception("Index out of range");
        return Rows[index];
    }
    set
    {
        if (index < 0 || index >= RowCount)
            throw new Exception("Index out of range");
         Rows[index]=value;
    }
}
2 references
public RowResult this[string KeyWord]...
```

Class String Indexer

```
2 references
public RowResult this[string KeyWord]
{
    get
    {
        foreach (var obj in Rows)
        {
            if (obj.Keyword == KeyWord)
                return obj;
        }
        throw new Exception("No data found");
    }
    set
    {
        int index = -1;
        for (var i = 0; i < RowCount; i++)
        {

            if (Rows[i].Keyword == KeyWord)
            {
                index = i;
                Rows[i] = value;
                break;
            }
        }
        if (index < 0 )
            throw new Exception("No data found");
    }
}
```

Source code

```
namespace NNDers10Ornek5
{
    0 references
    internal class Program
    {
        0 references
        static void Main(string[] args)
        {
            DbResult db = new DbResult(3);

            RowResult row1 = new RowResult { ID = 1, Keyword = "Sutun1", Value = 10 };
            db[0] = row1;
            RowResult row2 = new RowResult { ID = 2, Keyword = "Sutun2", Value = "Murat" };
            db[1] = row2;
            RowResult row3 = new RowResult { ID = 3, Keyword = "Sutun3", Value = 10.2 };
            db[2] = row3;

            Console.WriteLine("Sutun1 değeri :");
            Console.WriteLine(db["Sutun1"].Value.ToString());

            Console.WriteLine("\n2. sütün değeri :");
            Console.WriteLine(db[1].Value.ToString());

            Console.WriteLine("\nSutun3  değeri :");
            Console.WriteLine(db["Sutun3"].Value.ToString());
        }
    }
    11 references
    public class RowResult...
    4 references
    public class DbResult...
}
```

Çıktı

```
namespace NNDers10Ornek5
{
    0 references
    internal class Program
    {
        0 references
        static void Main(string[] args)
        {
            DbResult db = new DbResult(3);

            RowResult row1 = new RowResult { ID = 1, Keyword = "Sutun1", Value = 10 };
            db[0] = row1;
            RowResult row2 = new RowResult { ID = 2, Keyword = "Sutun2", Value = "Murat" };
            db[1] = row2;
            RowResult row3 = new RowResult { ID = 3, Keyword = "Sutun3", Value = 10.2 };
            db[2] = row3;
        }
    }
    11 reference
    public c
    4 references
```

```
C:\WINDOWS\system32\cmd.exe
Sutun1 değeri :
10

2. sütün değeri :
Murat

Sutun3  değeri :
10.2
Press any key to continue . . .
```

Ödev

- Veritabanı sisteminden sorgu sonucu dönen kayıtların nesne olarak erişilmesini sağlayan sınıfları kodlayınız.
- Temel sınıfdaki indexer iki boyutlu olsun.
- ilk boyut satırı ikinci boyut sutunu temsil etsin.
- ilk boyut hep rakam olurken ikinci boyut rakam veya metin ifadesi olabilsin.
- Örnek bir veritaban tablosu oluşturun ve en az 10 kayıt ve en az 10 sütün olsun.
- Sutun tipleri bir birinden farklı olsun.
- Oluşturacağınız sınıflardaki satır ve sütün sayısı sabit bir değer olarak girilmesin ve dinamik olarak değişebilsin.
- Select Sutun1, Sutun2, Sutun3 from Table ve Select * From Table sorgularının cevabını nesne döndörerek gösteriniz. Döndürülen kayıtlarda sütun tiplerinin farklı olduğunu ve tipinin ne olduğunu ekrana yazdırın. Tarih olan bir veriyi tarih formatında yazdırın.
- 25 Aralık 07.59'a kadar gönderilenler 10+2 puan, finala kadar gönderenler 10 puan (kayubmprogramlama1@gmail.com).