Indexer

Indexerlar aşağıdaki gibi tanımlanmaktadır.

Example

class Class1{

private int[] numbers = new int[3];

public int this[int index]{

get{

return numbers[index];

}

set{

numbers[index] = value;

}

}

}

Class1 obj = new Class1();

obj[0] = 1;

obj[1] = 2;

obj[2] = 3;

Console.WriteLine(obj[0]); *//1*

Console.WriteLine(obj[1]); *//2*

Console.WriteLine(obj[2]); *//3*

* class ve struct içinde indexer tanımlanabilir.
* Erişim belirteçlerine sahip olabilirler.
* Indexer içinde sadece get bloğu tanımlayarak readonly olmasını sağlayabilirsiniz.
* Indexerlar aşırı yüklenebilirler. Bu sayede aynı sınıfta bulunan birden fazla dizi indekslenebilmektedir.

Example

class Class1{

private string[] colors = new string[3];

private int[] numbers = new int[4];

public string this[int index]{

get{

return colors[index];

}

set{

colors[index] = value;

}

}

public int this[byte index]{

get{

return numbers[index];

}

set{

numbers[index] = value;

}

}

}

Class1 obj = new Class1();

obj[0] = "Red";

obj[1] = "Green";

obj[2] = "Blue";

obj[(byte)0] = 1;

obj[(byte)1] = 2;

obj[(byte)2] = 3;

obj[(byte)3] = 4;

Console.WriteLine(obj[0]); *//Red*

Console.WriteLine(obj[(byte)0]); *//1*

* Indexer aracılığı ile bir diziye değer aktarılırken sınır aşılabilir. Örneğin iki elemanlı bir dizinin üçüncü elemanının okunmak istenmesi ya da değer aktarılmaya çalışılması durumunda hata meydana gelir. Hataları önlemek için get ve set bloklarında kontroller yapılmalıdır.
* Çok boyutlu diziler için aşağıdaki gibi indexer hazırlanabilmektedir.

Example

class Class1{

private string[,] cities = new string[2, 3];

public string this[int row, int column]{

get{

return cities[row, column];

}

set{

cities[row, column] = value;

}

}

}