

ElementAt or ElementAtOrDefault

ElementAt, ElementAtOrDefault

- ✓ ElementAt and ElementAtOrDefault both are used (independently) to return an element from a specific index.
- ✓ If the element is not available at given index -
 - ✓ ElementAt will throw an exception.
 - ✓ But ElementAtOrDefault will return null.

```
0 references
class Program
{
    0 references
    static void Main(string[] args)
    {
        List<int> numbers = new List<int>() { 1, 2, 3, 4, 5, 6, 7, 8, 9, 10 };
        var ms = numbers.ElementAt(1);
        Console.ReadLine();
    }
}
```

```
0 references
static void Main(string[] args)
{
    List<int> numbers = new List<int>() { 1, 2, 3, 4, 5, 6, 7, 8, 9, 10 };
    var ms = numbers.ElementAt(10);
    Console.ReadLine();
}
```

Exception Unhandled

System.ArgumentOutOfRangeException: 'Index was out of range. Must be non-negative and less than the size of the collection. Parameter name: index'

[View Details](#) | [Copy Details](#)

▶ Exception Settings

ElementAtOrDefault

```

class Program
{
    static void Main(string[] args)
    {
        List<int> numbers = new List<int>() { 1, 2, 3, 4, 5, 6, 7, 8, 9, 10 };
        var ms = numbers.ElementAtOrDefault(3);
        Console.WriteLine();
    }
}

```

If it is not present you will get default value of the data type for primitive for non primitive you will get null

```

class Program
{
    static void Main(string[] args)
    {
        List<int> numbers = new List<int>() { 1, 2, 3, 4, 5, 6, 7, 8, 9, 10 };
        List<string> names = new List<string>() { "a", "b", "c" };
        var ms = numbers.ElementAtOrDefault(10);
        var ms1 = names.ElementAtOrDefault(10);
        Console.WriteLine();
    }
}

```

You cant use query syntax you can only use the mixed and you can also perform the validation

```

class Program
{
    static void Main(string[] args)
    {
        List<int> numbers = new List<int>() { 1, 2, 3, 4, 5, 6, 7, 8, 9, 10 };
        List<string> names = new List<string>() { "a", "b", "c" };
        var ms = numbers.Where(x => x > 3).ElementAtOrDefault(1);
        var ms1 = names.ElementAtOrDefault(10);
        var mixedS = (from n in numbers
                      select n).ElementAt(2);
        Console.WriteLine();
    }
}

```