Convert string[] to int[] in one line of code using LINQ

Ask Question

57

I have an array of integers in string form:

```
var arr = new string[] { "1", "2", "3", "4" };

I need to an array of 'real' integers to push it further:

void Foo(int[] arr) { .. }
```

I tried to cast int and it of course failed:

Foo(arr.Cast<int>.ToArray());

```
l can do next:

var list = new List<int>(arr.Length);
arr.ForEach(i => list.Add(Int32.Parse(i))); // maybe Convert.ToInt32() is better?
Foo(list.ToArray());

or

var list = new List<int>(arr.Length);
arr.ForEach(i => {
   int j;
   if (Int32.TryParse(i, out j)) // TryParse is faster, yeah
   {
     list.Add(j);
   }
}
Foo(list.ToArray());
```

but both looks ugly.

Is there any other ways to complete the task?



edited Nov 17 '14 at 21:54

asked Aug 19 '09 at 0:09



abatishchev

70.5k 70 267 397

- What's wrong with simply iterating through one collection, converting the value, and the adding it to the second? Seems pretty clear in intention to me. - Ed S. Aug 19 '09 at 0:11
- Otherwise, msdn.microsoft.com/en-us/library/73fe8cwf.aspx Ed S. Aug 19 '09 at 0:12
- Guess that I should have posted an answer as that was the accepted suggestion :-) - Ed S. Aug 19 '09 at 0:27
- Just FYI, I'm using this question here: stackoverflow.com/questions/1297325/... - Allen Rice Aug 19 '09 at 0:49

@Ed, yea, I would mark it as an answer! - abatishchev Aug 19 '09 at 11:20

6 Answers



Given an array you can use the Array.ConvertAll method:

523 int[] myInts = Array.ConvertAll(arr, s => int.Parse(s));

Home

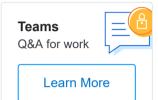
PUBLIC



Tags

Users

Jobs





Thanks to Marc Gravell for pointing out that the lambda can be omitted, yielding a shorter version shown below:

```
int[] myInts = Array.ConvertAll(arr, int.Parse);
```

A LINQ solution is similar, except you would need the extra ToArray call to get an array:

```
int[] myInts = arr.Select(int.Parse).ToArray();
```

edited Jun 9 '12 at 18:25

answered Aug 19 '09 at 0:15



Ahmad Mageed

77.8k 16 138 163

- 4 Nice. Didn't know that one. +1 spender Aug 19 '09 at 0:17
 - +1 for smallest footprint, even though LINQ is asked Marc Aug 19 '09 at 0:18
- 71 Actually, you don't need the lambda; ConvertAll(arr, int.Parse) is sufficient Marc Gravell ♦ Dec 8 '10 at 13:35
- 1 Lambda is needed in VB.Net 2010: uArray =
 Array.ConvertAll(sNums.Split(","), Function(i) UInteger.Parse(i)) BSalita
 Jan 15 '12 at 15:24 /*
- 1 @BSalita No, in VB.Net it's Array.ConvertAll(arr, AddressOf Integer.Parse) Slai May 3 '16 at 18:03



To avoid exceptions with .Parse , here are some .TryParse alternatives.

14

To use only the elements that can be parsed:

```
string[] arr = { null, " ", " 1 ", " 002 ", "3.0" };
     int i = 0;
     var a = (from s in arr where int.TryParse(s, out i) select i).ToArray
or
    var a = arr.SelectMany(s => int.TryParse(s, out i) ? new[] { i } : 
     int[0]).ToArray();
Alternatives using o for the elements that can't be parsed:
     int i;
     var a = Array.ConvertAll(arr, s => int.TryParse(s, out i) ? i : 0);
     0 }
or
     var a = arr.Select((s, i) => int.TryParse(s, out i) ? i : 0).ToArray
C# 7.0:
     var a = Array.ConvertAll(arr, s => int.TryParse(s, out var i) ? i : (
                                                                                                                                                                                 edited Aug 6 '18 at 15:09
                                                                                                                                                                                 answered May 4 '16 at 16:21
                                                                                                                                                                                                             15.6k 3 24 37
```

Thanks for this solution! - AngieM Nov 23 '16 at 14:43

The second solution: var a = Enumerable.Range(0, arr.Length).Where(i => int.TryParse(arr[i], out i)).ToArray(); just returns the indeces 0,1,2,... instead of the real values. What's the right solution here? — Beetee Jul 10 '17 at 11:20

Thanks @Beetee. Not sure what I was thinking with that. I replaced it with another alternative. – Slai Jul 10 '17 at 12:14

@Slai: Thanks. But what does new int[0]? When I have text, I don't get a 0 in my array... - Beetee Jul 11 '17 at 8:13

@Beetee <code>new int[0]</code> is an empty int array. The first two examples skip values that can't be parsed, and the last two examples use <code>0</code> for values that can't be parsed. — Slai Jul 11 '17 at 11:12



you can simply cast a string array to int array by:

12

var converted = arr.Select(int.Parse)



answered Jun 9 '12 at 13:07



A.Dara **689** 7 22

4 nice! thankyou. And in VB.Net Dim converted = arr.Select(addressof Integer.Parse) — Mafu Josh Mar 15 '13 at 12:25

✓



EDIT: to convert to array

30

int[] asIntegers = arr.Select(s => int.Parse(s)).ToArray();



This should do the trick:

```
var asIntegers = arr.Select(s => int.Parse(s));
```

edited Aug 19 '09 at 0:17

answered Aug 19 '09 at 0:11



Simon Fox

8,568 7 52 77

- 1 .ToArray() required to satisfy OP's question spender Aug 19 '09 at 0:13
- 1 change var to int[] and append .ToArray() if you need it as an int array Simon Fox Aug 19 '09 at 0:14



var asIntegers = arr.Select(s => int.Parse(s)).ToArray();

3

Have to make sure you are not getting an <code>IEnumerable<int></code> as a return





abatishchev

70.5k 70 267 397

answered Aug 19 '09 at 0:15



Rob

1,135 1 7 13



var list = arr.Select(i => Int32.Parse(i));

