Searching if value exists in a list of objects using Linq

Ask Question

190

46

Say I have a class Customer which has a property FirstName. Then I have a List.

Can LINQ be used to find if the list has a customer with Firstname = 'John' in a single statement.. how?

c# linq

asked Jul 1 '09 at 19:58

Tony_Henrich
16.6k 55 189 317

9 Answers

391

✓

LINQ defines an extension method that is perfect for solving this exact problem:

```
using System.Linq;
...
    bool has = list.An
```

make sure you reference System.Core.dll, that's where LINQ lives.

edited Nov 25 '12 at 18:22

answered Jul 1 '09 at 19:59

zvolkov
15.2k 8 61 74

15 Any is good, I wonder how many developers use Count when they should use Any? – RichardOD Jul 1 '09 at 20:19

11 You can also do a case insensitive search: cus => cus.FirstName.Equals("John", StringComparison.CurrentCultureIgnoreCase) – jmservera Jul 1 '09 at 20:40

1 I know this is an old question but why aren't we making use of the Exists method? Seeing as it is made to see if things exist. – Blackunknown Jul 9 '14 at 9:29

4 Because not all collections have Exists, and it does not take a lambda expression, but rather the object we are looking for itself. – zvolkov Jul 9 '14 at 12:10

@zvolkov, Any ideas why my resharper is suggesting I use bool has = list.All(cus => cus.FirstName != "John"); Is this more optimal ? – Gullu Dec 5 '18 at 15:55

zvolkov's answer is the perfect one to find out *if* there is such a

91

customer. If you need to *use* the customer afterwards, you can do:

```
Customer customer = lis
if (customer != null)
{
    // Use customer
}
```

I know this isn't what you were asking, but I thought I'd pre-empt a follow-on question :) (Of course, this only finds the *first* such customer... to find all of them, just use a normal `where` clause.)

answered Jul 1 '09 at 20:01

Jon Skeet
1089k 690 7946 8442

7 I'd point out that you might appreciate having done this later when it comes to debugging, if you find yourself suddenly curious which customer it was that fit the criteria. – mquander Jul 1 '09 at 21:55

1 Just bumping this answer up one cos I love the way SO community goes the extra step to add even more to the question/answer. – barneymc May 1 '14 at 16:42

1 thanks it helped me, but sometimes i just want to get `bool` result , so in that case `.Any` or `.FindIndex` is used here **which is fast** ? – stom Mar 6 '16 at 11:39

1 @stom: They're both O(N), basically... they're just linear searches. – Jon Skeet Mar 6 '16 at 12:09

bumping this up. I like the way how you use the syntax of list.FirstOrDefault instead of doing a list.Where().FirstOrDefault. – GunWanderer Feb 20 '17 at 19:05

20

One option for the follow on question (how to find a customer who might have any number of first names):

```
List<string> names = ne
bool has = customers.An
```

or to retrieve the customer from csv of similar list

```
string input = "John,Ma
List<string> names = in
customer = customers.Fi
```

edited Aug 9 '16 at 22:48
bold
350 3 13

answered Jul 13 '09 at 18:18
Mike Sackton
781 3 16

9

Using Linq you have many possibilities, here one without using lambdas:

```
//assuming list is a L
var hasJohn = (from cu
         where custome
         select custom
```

answered Jul 1 '09 at 20:28

jmservera
5,404 1 21 35

4

The technique i used before discovering `.Any()`:

```
var hasJohn = (from cu
       where customer.F
       select customer)
```

answered Dec 5 '12 at 22:44

Ian Boyd
121k 189 686 1013

4

```
customerList.Any(x=>x.
```

edited Jun 17 '13 at 7:02

Carlos Landeras
9,512 11 42 73

answered Jul 1 '09 at 20:01

Chris Brandsma
10.1k 5 40 56

This does not answer the question "if" such an entry exists; it merely enumerates the values if they do exist. An extra step is needed to determine if this enumeration is nonempty. – jason Jul 1 '09 at 20:04

Your linq is not correct, should be: from x in customerList ... – jmservera Jul 1 '09 at 20:42

Then change the Where to Any. Probably more philosophical for me. I rarely need to know if without caring which ones they are. @jmservera: you were right. I gave up LINQ a while back and now use Lambda exclusively. – Chris Brandsma Jul 1 '09 at 21:55

I don't mean to be pedantic when I say that using the lambda calls is still technically using LINQ. (In particular, you're using LINQ-to-Objects.) You're just using the method calls rather than language keywords. – Judah Gabriel Himan Jul 13 '09 at 18:16

How does this answer differ from the zvolkov's? – dotnetN00b May 15 '12 at 20:45

3

```
List<Customer> list =
Customer john = list.S
```

john will be null if no customer exists with a first name of "John".

answered Jul 1 '09 at 20:01

M4N
74.9k 40 191 246

2 That will throw an exception if *more than one* customer is

called John. – Jon Skeet Jul 1 '09 at 20:02

1 Thanks for the comment. I'll leave the answer as a partially correct example. – M4N Jul 1 '09 at 20:09

It's still valid in a scenario when you are sure there is 1 and you want an exception to be raised if more than one, so I think it is good that you didn't delete it. – RichardOD Jul 1 '09 at 20:21

1

Another possibility

```
if (list.Count(custome
 //bla
}
```

answered Jul 1 '09 at 20:14

Krassi
1,170 1 16 24

3 Its' preferable to use Any in this scenario. – RichardOD Jul 1 '09 at 20:22

Yeah, I didn't know, that Any() terminates if the element is found... – Krassi Jul 2 '09 at 0:03

0

Try this, I hope it helps you.

```
 if (lstCustumers.Any(
 {
     //TODO CODE
 }
```

answered Jun 13 '17 at 16:42

Fabio Stratotti
17 1

3 That's the same as the accepted answer from over 8 years ago. Please make sure your answer is unique among all the answers. – Tony_Henrich Jun 14 '17 at 22:46