Subtract a generic list from another

45

6

I am trying remove a list of firmIDs from one list from another. I don't really understand linq but I am pretty sure I need to use it.

```
List<Firm> firms = GetBusinessDevelopmentFirms(database);
List<Firm> trackedFirms = GetAllCLIFirmsBeingTrackedByUser();

var result = firms.Contains(i => trackedFirms.Contains(i.FirmID));
```

The last line doesn't work and the system says "unknown method Contains(?)" even though I have put "using System.Linq;" At the top of the class.

My idea was to remove a list of tracked firms from a list of all firms to find the untracked firms.

I hope this makes sense.

c# linq

edited Jan 18 '17 at 19:43
Stealth Rabbi
5,473 14 75 142

asked Sep 19 '13 at 13:18
Bobby
1,394 3 14 32

6 Answers

93

```
var result = firms.Except(trackedFirms); // returns all the firms except those in
trackedFirms
```

answered Sep 19 '13 at 13:21

13:24

Not that it really matters, but it should be noted that this is not Linq, but an existing method on List<T> – CodingIntrigue Sep 19 '13 at 13:24

Cheers for the heads up, if I don't have to use Linq that's fine. I will get my head around it one day. I will mark this as my answer when the site allows me to. Cheers all – Bobby Sep 19 '13 at 13:26

You are right. But please note that the compiler actually converts the linq queries to the corresponding Extension Methods. So no that much difference between them. Also, the last line of OP questions implies what he wants. – Alireza Sep 19 '13 at 13:27

2 @RGraham No, you just linked to the Linq method, you just linked to it from the section of extension methods on the `List<T>` page. `List<T>` has no instance `Except` method. If you select one of the overloads on that page you'll see the namespace they're defined in is `System.Linq` and that the first parameter is `IEnumerable<T>` not `List<T>` . – Servy Sep 19 '13 at 13:31

9

From your code above, I presume you are trying to get entries from Firms that have a corresponding item in TrackedFirms.

```
List<Firm> results = Firms.Where(f => TrackedFirms.Any(t => t.FirmId ==
f.FirmId)).ToList();
```

If on the other hand you want untracked firms, then it's :

```
List<Firm> results = Firms.Where(f => !TrackedFirms.Any(t => t.FirmId ==
f.FirmId)).ToList();
```

edited Jun 12 '18 at 7:09
Romil Kumar Jain
16.7k 7 43 81

answered Sep 19 '13 at 13:28
Martin Milan
5,440 1 26 41

I think this should works

```
var result = firms.Where(x => !trackedFirms.Any(y => x.FirmID == y.FirmID));
```

4

6,995 1 25 28

3

`Contains` is the native method of `List<T>` that expects you to pass in a `T`. You want `Where` instead.

```
var result = firms.Where(i => trackedFirms.Contains(i.FirmID));
```

If you expect `result` to be a `List<T>` then add `.ToList()` to the end of your `Where` LINQ expression.

answered Sep 19 '13 at 13:20
Eli Gassert
8,853 2 20 35

3 +1 for the .ToList(). I needed that! Cheers – Bobby Sep 19 '13 at 13:24

While my answer is most closely related to your question, I really prefer this answer to mine: stackoverflow.com/a/18895816/1795053 If you just add `.ToList()` to his, you'll get the same results. Or if you prefer to know the Intersection, then one of the `Intersect` results. It's more precise and more obvious what your goal is. I'd consider accepting one of their answers. Good luck! – Eli Gassert Sep 19 '13 at 13:26

1) This is the intersection of the two lists, not the subtraction of one from the other. 2) You're doing a linear search for each item, which is much slower than it could be if using a better algorithm. – Servy Sep 19 '13 at 13:33

1

The best approach would be `Except()` for you case. However, if the List object is different then other one, you can use `All()`

```
firms.Where(x=> trackedFirms.All(y => y.FirmId != x.FirmId)
```

answered Jun 26 '18 at 14:33
Mehmet Taha Meral
628 6 16

Join Stack Overflow to learn, share knowledge, and build your career.

Sign up with email Sign up with Google Sign up with Facebook

[Comparing two Lists and returning the distinct values and the differences]

edited May 23 '17 at 11:47
Community ♦
1 1

answered Sep 19 '13 at 13:21
Steven Wood
905 3 16 40

1 Intersect isn't set subtraction. – Servy Sep 19 '13 at 13:33