LINQ

...

Language **IN**tegrated **Q**ueries

But in Racket

#lang trackette

What is LINQ?

Language-Integrated Query (LINQ) is the name for a set of technologies based on the integration of query capabilities directly into the C# language.

With LINQ, a query is a first-class language construct, just like classes, methods, events.

LINQ Grammar

query-expression ::= from-clause query-body

query-body ::= query-body-clause* final-query-clause query-continuation?

query-body-clause ::= (from-clause | join-clause | let-clause | where-clause | orderby-clause)

from-clause ::= from itemName in srcExpr

join-clause ::= join itemName in srcExpr on keyExpr equals keyExpr (into itemName)

let-clause ::= let itemName = selExpr

where-clause ::= where predExpr

orderby-clause ::= orderby (keyExpr (ascending | descending))*

final-query-clause ::= (select-clause | groupby-clause)

select-clause ::= select selExpr

groupby-clause ::= group selExpr by keyExpr

query-continuation ::= into itemName query-body

Approaches

1. Naively convert (`from x in src …`) to a `for`/`list`

2. Worked with single sequences, but not with multiple `from`'s, combining multiple sequences together

3. The Real LINQ™



The Fellowship of The Ring (Naive Approach)

- Convert `(from x in src …)` to `(for/list ([x src]) …)`.

- But what about `where`, `orderby`?

- How do these changes get propagated to the next expression?

The Two Towers

- `from` turns into an application of all its expressions onto the source expression.

- Each query expression turns into a function expecting this source expression.

- What about multiple `from`'s?

Example translation

```
(from x in '(1 2 3)
   (where (>= x 2))   =>
   (select (+ x 3)))
```

```
((lambda (ls)
   (sequence-map
    (lambda (x) (+ x 3)) ls)
    ((lambda (ls)
       (sequence-filter
        (lambda (x) (>= x 2)) ls))
      '(1 2 3)))
```

The Return of the King (LINQ)

- LINQ transforms variables into structure references.

- We did something similar by translating variables into function applications (`cdr (cdr …)`) onto pairs.

LINQ Example

```
from i in arr1
from j in arr2
from k in arr1
from l in arr2
select i + j + k;
```

```
arr1.SelectMany(i => arr2, (i, j) => new tempStruct1(i = i, j = j))
    .SelectMany(id1 => arr1, (id1, k) => new tempStruct2(id1 = id1, k = k))
    .SelectMany(id2 => arr2, (id2, l) => (((id2.id1.i + id2.id1.j) + id2.k) + l))
```