

RINQ

Krzysztof Kosyl

Toruń 16 kwietnia 2008

1 LINQ

2 Metaprogramowanie

- Eval
- DSL
- Metaprogramowanie w Ruby

3 RINQ

- Operatory zapytań

LINQ

Language INtegrated Query

- Obiekty
- XML
- SQL

```
int[] numbers = { 5, 4, 1, 3, 9, 8, 6, 7, 2, 0 };
```

```
var q =  
    from n in numbers  
    where n < 5  
    select n;
```

```
//XElement contacts = XElement.Parse("...")  
XElement doc = XElement.Load(@"numbers.xml");  
var q = new XElement("numbers",  
    from n in doc.Elements("number")  
    where (int) n < 5  
    select new XElement("number", n)  
);
```

```
Database db = Database(...)  
var q =  
    from n in db.numbers  
    where n.number < 5  
    select new {n.number};
```

Metaprogramowanie

Programy piszące programy

```
name = 'plus'  
op = '+'  
a = "def #{name}(a, b) a #{op} b end"  
eval(a)  
plus(1,2)
```

DSL

Domain Specific Language

- znaki drogowe
- notacja muzyczna
- Makefile
- RubyOnRails ActiveRecord

Symbole

'long_and_eazy_name'

:long_and_eazy_name

Atrybuty - settery, gettery

```
class Cos
  def a
    @a
  end
  def a=(value)
    @a = value
  end
end
```

```
class Cos
  attr_accessor :a
end
```

Pomijanie nawiasów w wywołaniu funkcji

```
c = Cos.new()  
c.a=(7)  
puts c.a()
```

```
funkcja({'a'=>'Ala', b=>'Beata'})
```

```
inna_funkcja(3.14, {'a'=>'Ala'})
```

```
c = Cos.new  
c.a = 7  
puts c.a
```

```
funkcja 'a'=>'Ala', b=>'Beata'
```

```
inna_funkcja 3.14, 'a'=>'Ala'
```

Otwarte moduły

```
module Kernel
  def query()
    puts 'this is query'
  end
end
```

Otwarte klasy

```
class Fixnum
  def hours
    self * 3600
  end
  alias hour hours
end
```

```
Time.mktime(2006, 01, 01) + 14.hours
```

Otwarte obiekty

```
a = 'hello'
```

```
class <<a  
  def to_s  
    "<#{self}>"  
  end  
end
```

Brakujące metody

```
class Cos
  def method_missing(method_name, *args)
    method_name = method_name.to_s
    if method_name[0..3] == 'say_'
      puts method_name[4..-1]
    else
      raise NoMethodError, "'#{method_name}' in #{self}"
    end
  end
end
```

Domknięcia

```
def two_times()  
  if block_given?  
    yield  
    yield  
  end  
end  
two_times { puts 'Hello' }
```

```
[1,2,3,4].delete_if { |a| a % 2 == 0 }
```

RINQ

Ruby Integrated Query Language

Operatory zapytań

```
customernames = customers.  
  qwhere    { |c| c.address.city == "Torun" }.  
  qselect   { |c| { :firstname => c.firstname,  
                  :lastname => c.lastname } }.  
  qorderby  { |c| c.lastname }
```

Wyrażenia zapytań

```
customerfirstnames = query do
  qfrom :c => customers
  qwhere c.lastname == "Mitchel"
  qselect c.firstname
end
```

`http://www.xaop.com/blog/2007/10/07/
video-how-to-create-a-domain-specific-language-/`