Introdução a Programação Funcional com F#

Fabio Galuppo, M.Sc.

http://fabiogaluppo.com e http://simplycpp.com/

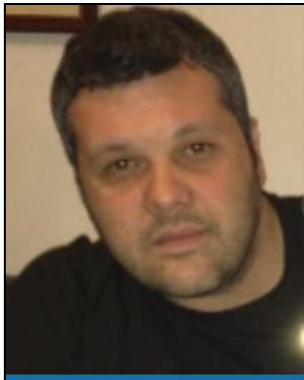
fabiogaluppo@acm.org

@FabioGaluppo

Microsoft MVP Visual Studio and Development Technologies

https://mvp.microsoft.com/en-us/PublicProfile/9529

http://bit.ly/8o_SeCoT



Award Categories
Visual Studio and Development Technologies

First year awarded:
2002

Number of MVP Awards:
13

Fabio Razzo Galuppo, M.Sc.

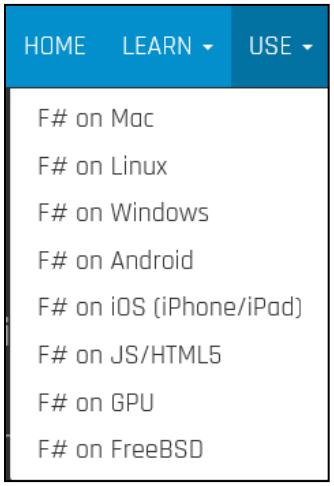
Novembro 1973

- Mestrado em Engenharia Elétrica (Universidade Presbiteriana Mackenzie)
  - Ciência da Computação - Inteligência Artificial
- Por mais de 10 anos premiado com Microsoft MVP em Visual C++
- Engenheiro de Software (Programador)
- Matemática Aplicada
- Linguagens de programação prediletas:
  - C++
  - F#
  - Haskell
- Rock'n'Roll
  - E boa música em geral
- http://fabiogaluppo.com
- https://twitter.com/FabioGaluppo
- https://github.com/fabiogaluppo
- http://simplycpp.com

F#

F# is a mature, open source, cross-platform, functional-first programming language. It empowers users and organizations to tackle complex computing problems with simple, maintainable and robust code.

F# runs on Linux, Mac OS X, Android, iOS, Windows, GPUs, and browsers. It is free to use and is open source under an OSI-approved license.



http://fsharp.org/

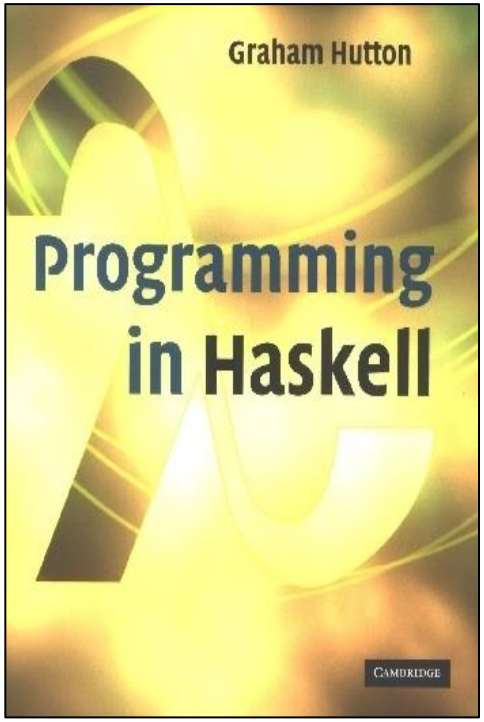
Fabio Galuppo - http://member.acm.org/~fabiogaluppo - fabiogaluppo@acm.org

Programação Funcional

What is a Functional Language?

Opinions differ, and it is difficult to give a precise definition, but generally speaking:

- Functional programming is style of programming in which the basic method of computation is the application of functions to arguments;
- A functional language is one that supports and encourages the functional style.



http://www.cs.nott.ac.uk/~gmh/book.html

Fabio Galuppo - http://member.acm.org/~fabiogaluppo - fabiogaluppo@acm.org

F# é *functional-first*

- Programação Funcional

```
let sumOnlyPositivesFunctional (xs : int list) =
    xs |> List.filter (fun x -> x > 0)
       |> List.sum
```

- Programação Imperativa

```
let sumOnlyPositivesImperative (xs : int list) =
    let mutable acc = 0
    for x in xs do
        if (x > 0) then acc <- acc + x
    acc
```

- Orientação a Objetos

Fabio Galuppo - http://member.acm.org/~fabiogaluppo - fabiogaluppo@acm.org

Exemplo #1

$$\lim_{x \to 2}(3 * x^2)$$

```
//Limit of f when x approaches x0
let Limit f x0 =
    let threshold = 0.0001
    let left  = f (x0 - threshold)
    let right = f (x0 + threshold)
    let estimate = (left + right) / 2.
    let threshold = 0.01
    if (abs (left - right) < threshold) then
        Some estimate
    else
        None

let displayLimit f x0 =
    let result = Limit f x0
    match result with
    | Some (value) -> printfn "The limit of f when x approaches %f is %f" x0 value
    | None -> printfn "The limit does not exist"
```

```
displayLimit (fun x -> 3. * (x ** 2.)) 2.
```

```
The limit of f when x approaches 2.000000 is 12.000000
```

Fabio Galuppo - http://member.acm.org/~fabiogaluppo - fabiogaluppo@acm.org

Exemplo #2

```
let ld = lastDayOfMonth(month, year)
let dw = 7 - dayOfWeek(month, year)
//days of the month by week
let xss = [[1..dw];
           [dw + 1..dw + 7];
           [dw + 8..dw + 14];
           [dw + 15..dw + 21];
           [dw + 22..min(ld, dw + 28)];
           [min(ld + 1, dw + 29)..ld]]
```

```
          Mai
 D  S  T  Q  Q  S  S
 1  2  3  4  5  6  7
 8  9 10 11 12 13 14
15 16 17 18 19 20 21
22 23 24 25 26 27 28
29 30 31
```

```
//build calendar as string
let h = xss |> List.head
let ts = xss |> List.tail
let months = [|"Jan"; "Fev"; "Mar"; "Abr"; "Mai"; "Jun"; "Jul"; "Ago"; "Set"; "Out"; "Nov"; "Dez"|]
let title = [sprintf "%12s" months.[month - 1]; " D  S  T  Q  Q  S  S"]
let firstRow = [h |> stringFormat (right 21)]
let remainingRows = ts |> List.collect (fun xs -> [xs |> stringFormat (left 21)])
remainingRows |> List.append firstRow
              |> List.append title
              |> toStringLn
```

Fabio Galuppo - http://member.acm.org/~fabiogaluppo - fabiogaluppo@acm.org

Exemplo #3

```
let createRnd () = { rnd = new Random() }
let createRndWithSeed (seed) = { rnd = new Random(seed) }
```

```
//[0.0, 1.0)
let uniform (r : Rnd) = r.rnd.NextDouble()
//[0, n)
let uniformInt (r : Rnd) n =
    precondition (n > 0) "less or equal than 0"
    r.rnd.Next(n)
```

```
//[n, m)
let uniformRange (r : Rnd) n m =
    precondition (m > n) "invalid range"
    n + uniform r * (m - n)
//[n, m)
let uniformIntRange (r : Rnd) n m =
    precondition (m > n) "invalid range"
    precondition (int64 (m - n) < int64 Int32.MaxValue) "invalid range"
    n + uniformInt r (m - n)
```

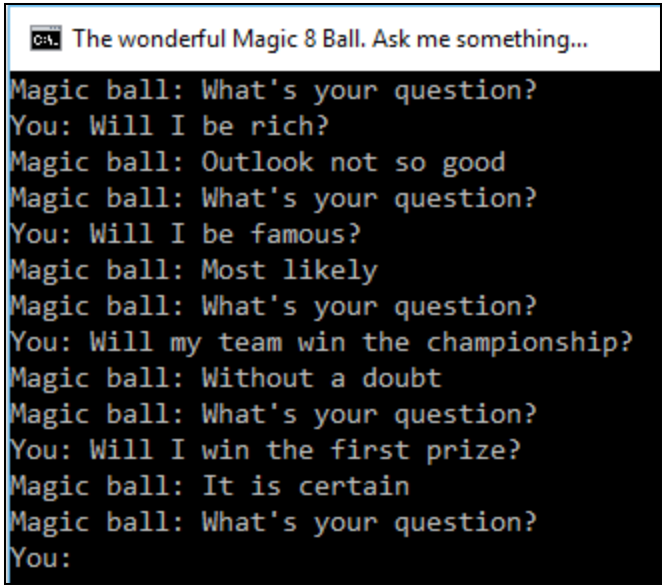
```
let gaussian (r : Rnd) mu (* mean *) sigma (* std dev *) =
    //Polar form of the Box-Muller transform
    let gaussian1 ()  =
        let rec loop x h =
            if (h >= 1. || h = 0.) then
                let x = uniformRange r -1. 1.
                let y = uniformRange r -1. 1.
                let h = x * x + y * y
                loop x h
            else
                x * Math.Sqrt(-2. * Math.Log(h) / h)
        loop 0. 0.
    mu + sigma * gaussian1 ()
```

```
gaussian: [(1, 14); (2, 121); (3, 104); (4, 11)]
----------
uniform : [(1, 58); (2, 71); (3, 56); (4, 65)]
```

Fabio Galuppo - http://member.acm.org/~fabiogaluppo - fabiogaluppo@acm.org

Exemplo #4

```
let shuffle (r : Rnd) (xs : 'a array) =
    precondition (xs <> null) "array is null"
    let n = xs |> Array.length
    for i = 0 to n - 1 do
        let j = i + uniformInt r (n - i)
        let temp = xs.[i]
        xs.[i] <- xs.[j]
        xs.[j] <- temp
```




```
let rec loop (f : unit -> int) =
    printfn "Magic ball: What's your question?"
    printf "You: "
    let ask = Console.ReadLine()
    if String.IsNullOrEmpty(ask) then ()
    else
        printfn "Magic ball: %s" magic8BallAnswers.[f ()]
        loop f
let n = magic8BallAnswers |> Array.length
let now = DateTime.Now
let r = createRndWithSeed (now.Millisecond + now.Second * 1000)
magic8BallAnswers |> shuffle r
loop (fun () -> uniformIntRange r 0 n)
```

Fabio Galuppo - http://member.acm.org/~fabiogaluppo - fabiogaluppo@acm.org

Exemplo #5

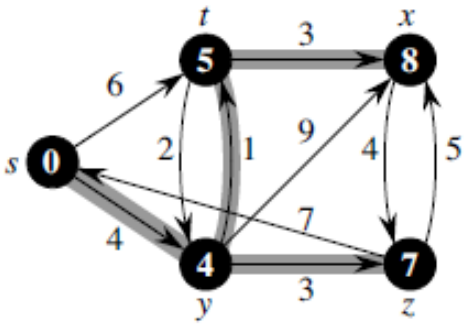
```
//relax procedure
let relax u v w =
     //w == weight u v
     let dist = shortest.[u] + w
     if dist < shortest.[v] then
        shortest.[v] <- dist
        pred.[v] <- u
```

```
["y -> t"; "t -> x"; "s -> y"; "y -> z"]
y -> t
t -> x
s -> y
y -> z
```

```
let directedEdges =
    //u -> v : w == u, v, w
    [
        (s, t, 6); (s, y, 4);
        (t, x, 3); (t, y, 2);
        (x, z, 4);
        (y, t, 1); (y, z, 3); (y, x, 9);
        (z, x, 5); (z, s, 7)
    ]

let sp = directedEdges |> bellman_fordSP s
```

```
//BELLMAN-FORD algorithm
for i = 0 to n - 1 do
    shortest.[i] <- inf
    pred.[i] <- nil
shortest.[source] <- 0
for i = 1 to n - 1 do
    for (u, v, w) in weightedDirectedEdges do
        relax u v w
pred |> Array.mapi (fun i p -> (p, i, shortest.[i]))
     |> Array.filter (fun (x, _, _) -> x <> nil)
     |> Array.toList
```



Fabio Galuppo - http://member.acm.org/~fabiogaluppo - fabiogaluppo@acm.org

Exemplo #6

```
open FSharp.Data

type Wiki_1982_FIFA_World_Cup = HtmlProvider<"https://en.wikipedia.org/wiki/1982_FIFA_World_Cup">
let instance1982FWC = Wiki_1982_FIFA_World_Cup.GetSample()
let retrospective1982 = instance1982FWC.Tables.``FIFA retrospective ranking``
```

```
Ranking Table:
R  Team                G  P  W  D  L GF GA  GD Pts.
 1 Italy              1/C  7  4  3  0 12  6  +6  11
 2 West Germany       2/B  7  3  2  2 12 10  +2   8
 3 Poland             1/A  7  3  3  1 11  5  +6   9
 4 France             4/D  7  3  2  2 16 12  +4   8
 5 Brazil             6/C  5  4  0  1 15  6  +9   8
 6 England            4/B  5  3  2  0  6  1  +5   8
 7 Soviet Union       6/A  5  2  2  1  7  4  +3   6
 8 Austria            2/D  5  2  1  2  5  4  +1   5
 9 Northern Ireland   5/D  5  1  3  1  5  7  -2   5
10 Belgium            3/A  5  2  1  2  3  5  -2   5
11 Argentina          3/C  5  2  0  3  8  7  +1   4
12 Spain              5/B  5  1  2  2  4  5  -1   4
13 Algeria              2  3  2  0  1  5  5   0   4
14 Hungary              3  3  1  1  1 12  6  +6   3
15 Scotland             6  3  1  1  1  8  8   0   3
16 Yugoslavia           5  3  1  1  1  2  2   0   3
17 Cameroon             1  3  0  3  0  1  1   0   3
18 Honduras             5  3  0  2  1  2  3  -1   2
19 Czechoslovakia       4  3  0  2  1  2  4  -2   2
20 Peru                 1  3  0  2  1  2  6  -4   2
21 Kuwait               4  3  0  1  2  2  6  -4   1
22 Chile                2  3  0  0  3  3  8  -5   0
23 New Zealand          6  3  0  0  3  2 12 -10   0
24 El Salvador          3  3  0  0  3  1 13 -12   0
```

R	Team	G	P	W	D	L	GF	GA	GD	Pts.
1	Italy	1/C	7	4	3	0	12	6	+6	11
2	West Germany	2/B	7	3	2	2	12	10	+2	8
3	Poland	1/A	7	3	3	1	11	5	+6	9
4	France	4/D	7	3	2	2	16	12	+4	8
Eliminated in the second group stage										
5	Brazil	6/C	5	4	0	1	15	6	+9	8
6	England	4/B	5	3	2	0	6	1	+5	8
7	Soviet Union	6/A	5	2	2	1	7	4	+3	6
8	Austria	2/D	5	2	1	2	5	4	+1	5
9	Northern Ireland	5/D	5	1	3	1	5	7	−2	5
10	Belgium	3/A	5	2	1	2	3	5	−2	5
11	Argentina	3/C	5	2	0	3	8	7	+1	4
12	Spain	5/B	5	1	2	2	4	5	−1	4
Eliminated in the first group stage										
13	Algeria	2	3	2	0	1	5	5	0	4
14	Hungary	3	3	1	1	1	12	6	+6	3
15	Scotland	6	3	1	1	1	8	8	0	3
16	Yugoslavia	5	3	1	1	1	2	2	0	3
17	Cameroon	1	3	0	3	0	1	1	0	3
18	Honduras	5	3	0	2	1	2	3	−1	2
19	Czechoslovakia	4	3	0	2	1	2	4	−2	2
20	Peru	1	3	0	2	1	2	6	−4	2
21	Kuwait	4	3	0	1	2	2	6	−4	1
22	Chile	2	3	0	0	3	3	8	−5	0
23	New Zealand	6	3	0	0	3	2	12	−10	0
24	El Salvador	3	3	0	0	3	1	13	−12	0

https://en.wikipedia.org/wiki/1982_FIFA_World_Cup

Fabio Galuppo - http://member.acm.org/~fabiogaluppo - fabiogaluppo@acm.org

Exemplo #6

https://blogs.msdn.microsoft.com/dsyme/2013/01/30/twelve-f-type-providers-in-action/

- Doze type providers em ação
  - SQL, CSV, XML, JSON, WMI, OData, Hadoop/Hive, World Bank, Freebase, R, WSDL, TypeScript

```
Top 10 in Goal Difference:
Brazil                9
Italy                 6
Poland                6
Hungary               6
England               5
France                4
Soviet Union          3
West Germany          2
Austria               1
Argentina             1
```

```
Top 10 in Goals For:
France               16
Brazil               15
Italy                12
West Germany         12
Hungary              12
Poland               11
Scotland              8
Argentina             8
Soviet Union          7
England               6
```

http://fsharp.github.io/FSharp.Data/

Fabio Galuppo - http://member.acm.org/~fabiogaluppo - fabiogaluppo@acm.org

Exemplo #7

```
type PingAgent() as self =
    let receiver (inbox : MailboxProcessor<Msg * IAgent<Msg>>) =
        let rec loop () =
            async {
                let! (msg, actor) = inbox.Receive()
                match msg with
                | Pong(n) ->
                    printfn "[%d] Ping received pong : %d" (tid()) n
                    if n > 1 then
                        self |> send (Ping (n - 1)) actor
                        return! loop ()
                    else
                        printfn "[%d] Ping finished" (tid())
                        self |> send Stop actor
                        return ()
                | _ -> return! loop ()
            }
        loop ()
    let inbox = MailboxProcessor.Start(receiver)
    interface IAgent<Msg> with
          member this.Inbox = inbox
          member this.Send(msg, actor) = actor.Inbox.Post(msg, upcast this)
```

```
let ping = new PingAgent()
let pong = new PongAgent()

ping |> send (Ping 3) pong
```

Fabio Galuppo - http://member.acm.org/~fabiogaluppo - fabiogaluppo@acm.org

Exemplo #7

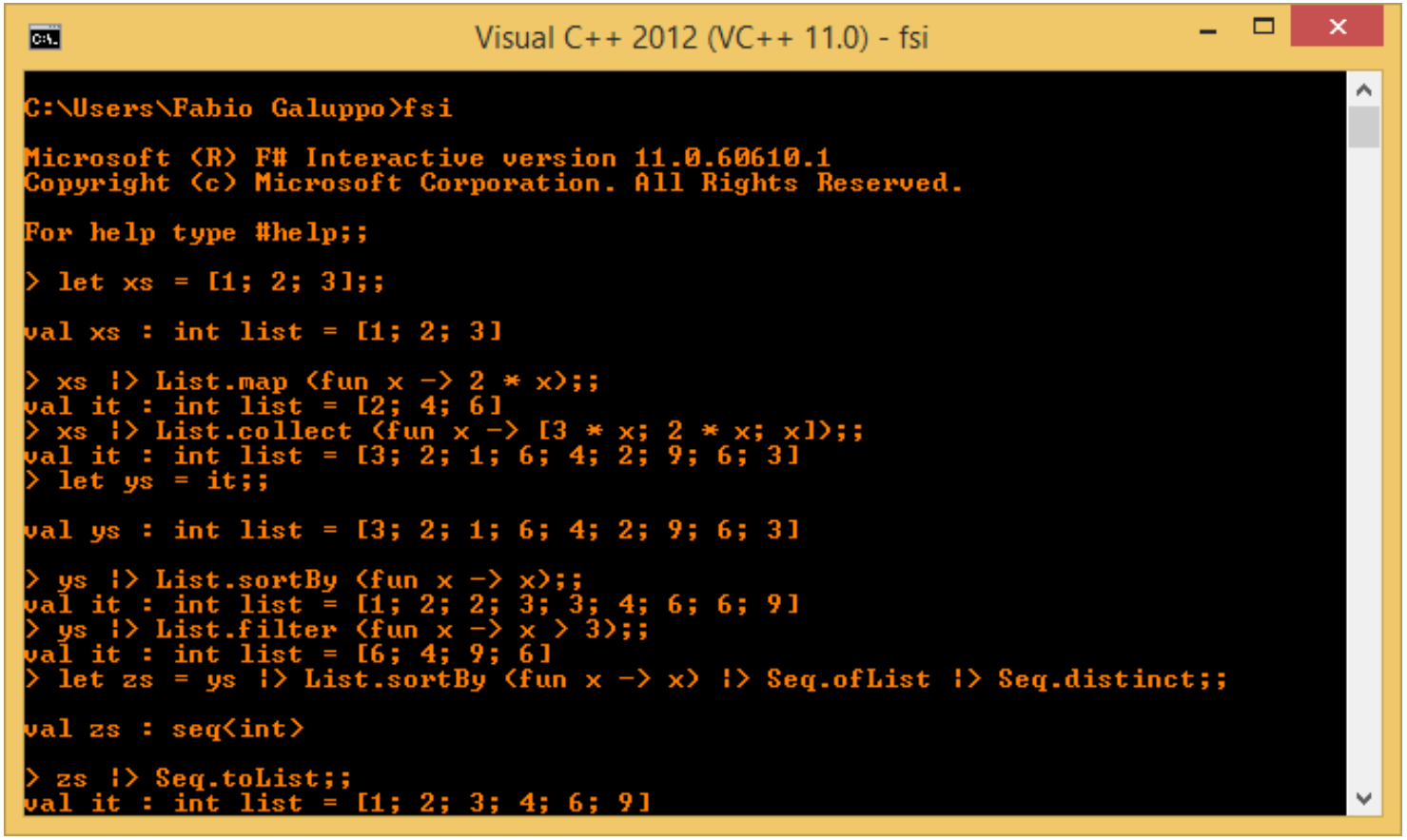
```
type PongAgent() as self =
    let receiver (inbox : MailboxProcessor<Msg * IAgent<Msg>>) =
        let rec loop () =
            async {
                let! (msg, actor) = inbox.Receive()
                match msg with
                | Stop ->
                    printfn "[%d] Pong finished" (tid())
                    return ()
                | Ping(n) ->
                    printfn "[%d] Pong received ping : %d" (tid()) n
                    self |> send (Pong n) actor
                    return! loop ()
                | _ -> return! loop ()
            }
        loop ()
    let inbox = MailboxProcessor.Start(receiver)
    interface IAgent<Msg> with
          member this.Inbox = inbox
          member this.Send(msg, actor) = actor.Inbox.Post(msg, upcast this)
```

```
[5] Pong received ping : 3
[3] Ping received pong : 3
[4] Pong received ping : 2
[3] Ping received pong : 2
[3] Pong received ping : 1
[4] Ping received pong : 1
[4] Ping finished
[3] Pong finished
```

Fabio Galuppo - http://member.acm.org/~fabiogaluppo - fabiogaluppo@acm.org

F# e LINQ (C#)

F# : `map`, `collect`, `filter`, `sortBy`, and `distinct`



C# : `Select`, `SelectMany`, `Where`, `OrderBy`, and `Distinct`

Fabio Galuppo - http://member.acm.org/~fabiogaluppo - fabiogaluppo@acm.org

Exemplo #8 (Bônus)

```
type RockOn = HtmlProvider<"https://programarockon.com.br/">
```

```
xs.Elements()
   |> List.filter (fun x -> x.Name() = "div" && x.HasAttribute("id", "page"))
   |> List.collect (fun x -> x.Descendants() |> Seq.toList)
   |> Seq.filter (fun x -> x.Name() = "div" && x.HasAttribute("id", "content"))
   |> Seq.collect (fun x -> x.Descendants())
   |> Seq.filter (fun x -> x.Name() = "div" && x.HasAttribute("id", "primary"))
   |> Seq.collect (fun x -> x.Descendants())
   |> Seq.filter (fun x -> x.Name() = "main" && x.HasAttribute("id", "main"))
   |> Seq.collect (fun x -> x.Descendants())
   |> Seq.filter (fun x -> x.Name() = "article")
   |> Seq.collect (fun x -> x.Descendants())
   |> Seq.filter (fun x -> x.Name() = "div" && x.HasAttribute("class", "entry-content"))
   |> Seq.collect (fun x -> x.Descendants())
   |> Seq.filter (fun x -> (x.Name() = "p" || x.Name() = "div") &&
                           (x.InnerText() |> hasProgram || x.InnerText() |> hasBand))
   |> Seq.map (fun x -> x.InnerText())
```

```
let uniqueBands = tracks |> Seq.collect (fun x -> bands x)
                         |> Seq.map (fun x -> x |> textWithoutPeriod)
                         |> Seq.map (fun x -> x.ToUpper())
                         |> Seq.sort
                         |> Seq.distinct
```

Fabio Galuppo - http://member.acm.org/~fabiogaluppo - fabiogaluppo@acm.org

Exemplo #8 (Bônus)

```
let dumpPages n =
    seq {
        for i = 1 to n do
            let instanceRockOn = RockOn.Load("https://programarockon.com.br/page/" + (string i) + "/")
            let xs = instanceRockOn.Lists.Html.Body()
            yield dumpPage xs
    }

let allPages = dumpPages NUMBER_OF_PAGES
```

```
Programa #39
Some
  ("Programa Rock ON #39",
   "Faixas: Last in Line, Thundermother, Grand Magus, Alia Tempora, David Coverdale, Misbehaviour,
 Bullseye, Meghavory, Blues Pills, Don Airey, Abysmal Dawn, Dawn of Demise, Iron Maiden.")
```

```
# of Bands = 632
 List of Bands:
[220 VOLT] [3 INCHES OF BLOOD] [4ARM] [ABBA (YNGWIE MALMSTEEN)] [ABHORRENT] [ABHORRENT DECIMATION]
 [ABORTED] [ABRASION] [ABYSMAL DAWN] [AC/DC] [ACCEPT] [ACE FREHLEY] [ACIDO] [ACLLA] [ACT OF DEFIAN
CE] [ADRENALINE RUSH] [AETERNA] [AGENTZ] [AJNA] [ALIA TEMPORA] [ALICE COOPER] [ALKALOID] [ALMANAC]
 [AMBUSH] [AMON AMARTH] [AMORPHIS] [ANCESTTRAL] [ANCIENT BARDS] [ANGRA] [ANGRY] [ANTHARES] [ANTHRA
X] [AOR] [APE SKULL] [APOCALYPTICA] [ARAÑA] [ARCH ENEMY] [ARCH/MATHEOS] [ARETHA FRANKLIN (LITTLE C
AESAR)] [ARMAHDA] [ARMORED SAINT] [ART OF ANARCHY] [ARTHEMIS] [ARTILLERY] [ASIA] [ASPHYX] [ASSAULT
ER] [ASTARTE] [ATTIC DEMONS] [ATTOMIC SOLDIER] [ATTRACHTA] [AUTOPSY] [AVANTASIA] [AVATAR] [AXEL RU
DI PELL] [BABYLON A.D] [BARREN CROSS] [BATTLE BEAST] [BATTLECROSS] [BEAUTIFUL SIN] [BEAUVOIR/FREE]
 [BELLA UTOPIA] [BELPHEGOR] [BENIGHTED] [BIG BALL] [BIGFOOT] [BIOCANCER] [BIONIC ORIGIN] [BITCH] [
BLACK SABBATH] [BLACK STATE HIGHWAY] [BLACK STONE CHERRY] [BLACK TORA] [BLACKNING] [BLACKSTAR RIDE
RS] [BLAZE BAYLEY] [BLAZE OUT] [BLAZING DOG] [BLIND GUARDIAN] [BLOODBATH] [BLOODBOUND] [BLOODGOOD]
 [BLOODRED HOURGLASS] [BLUES PILLS] [BOLT THROWER] [BONAFIDE] [BONFIRE] [BOREALIS] [BRAVEHEART] [B
RIAN MAY] [BRIDE] [BRUCE DICKINSON] [BULLSEYE] [BURNING BLACK] [BURNING POINT] [BURNING WITCH] [CA
COPHONY] [CAIN'S OFFERING] [CAN OF WORMS] [CANCER] [CANDLEMASS] [CANNIBAL CORPSE] [CARCASS] [CAUTE
RIZATION] [CAVERA] [CHAOS SYNOPSIS] [CHASTAIN] [CHEERS LEADERS] [CHILDREN OF BODOM] [CHRIS SQUIRE
(HOMENAGEM)] [CHRISTOPHER CROSS (SAXON)] [CHRONOSPHERE] [CIRCLE II CIRCLE] [CIRCLE OF INDIFFERENCE
] [CIRCLE OF INFINITY] [CIRCUS MAXIMUS] [CLIMATIC TERRA] [CONCEPT OF HATE] [CRADLE OF FILTH] [CRAN
IOTOMY] [CRIMSON GLORY] [CROSSROCK] [CROWN OF GLORY] [CROWN OF THORNS] [CRUENTA LACRYMIS] [CRYPTOP
SY] [CURSED SLAUGHTER] [CUT UP] [D.A] [DAFT PUNK (HALESTORM)] [DAGOBA] [DANCING FLAME] [DANGER ZON
E] [DARE] [DARK TRANQUILITY] [DARKANE] [DARKING] [DAVID COVERDALE] [DAWN OF DEMISE] [DE LA MUERTE]
```

Fabio Galuppo - http://member.acm.org/~fabiogaluppo - fabiogaluppo@acm.org

non-profit books and tutorials

cross-platform community data science

F# Software Foundation

commercial support open-source contributions

machine learning www.fsharp.org web and cloud

consulting user groups research

http://fsharp.org/

Community chat and Q&A

- **#fsharp** on Twitter
- **StackOverflow** F# tag

Open source on GitHub

- **Visual F#** repo github.com/Microsoft/visualfsharp
- **F# Compiler** and core libraries github.com/fsharp
- **F# Incubation** project space github.com/fsprojects
- **FsLab** Organization repository github.com/fslaborg

http://fsharp.org/

Livros e Recursos

fsharp.org/about/learning.html

Exemplos

http://www.fssnip.net/

Recent snippets

- Message Passing: Ping Pong

  Message Passing sample inspired by Erlang Ping Pong from here: http://erlang.org/doc/getting_started/conc_prog.html

  1 people like this
  Like the snippet!
  Posted: 12 hours ago by Fabio Galuppo

- Async ParallelWithThrottle

  Implementation of ParallelWithThrottle to limit the number of

- Concurrent Memoization with Timeout

  This is variant of http://www.fssnip.net/sA that is having a time-out. You may want to use this if you e.g. cache a database query result or other mutable data source.

  0 people like this
  Like the snippet!
  Posted: 3 days ago by Tuomas Hietanen

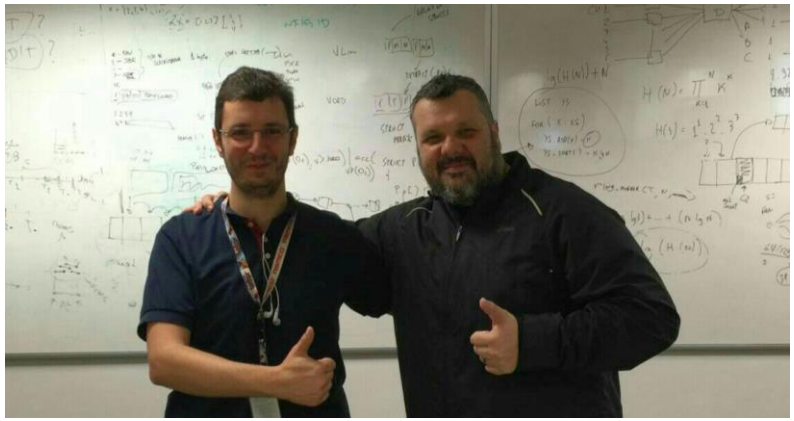
- Magic 8 Ball

Se quiser saber mais sobre:

C++
Programação Genérica
Iterators
Policy-based Design
Algoritmos
...

visite:

www.simplycpp.com



http://www.simplycpp.com

Fabio Galuppo - http://member.acm.org/~fabiogaluppo - fabiogaluppo@acm.org

Introdução a Programação Funcional com F#

Fabio Galuppo, M.Sc.

http://fabiogaluppo.com e http://simplycpp.com/

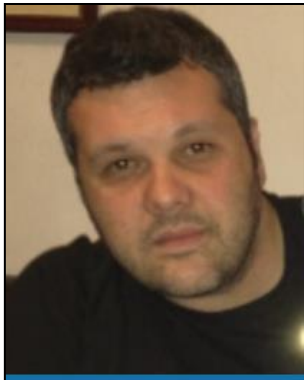
fabiogaluppo@acm.org

@FabioGaluppo

Microsoft MVP Visual Studio and Development Technologies

https://mvp.microsoft.com/en-us/PublicProfile/9529

http://bit.ly/8o_SeCoT



Award Categories
Visual Studio and Development Technologies

First year awarded:
2002

Number of MVP Awards:
13