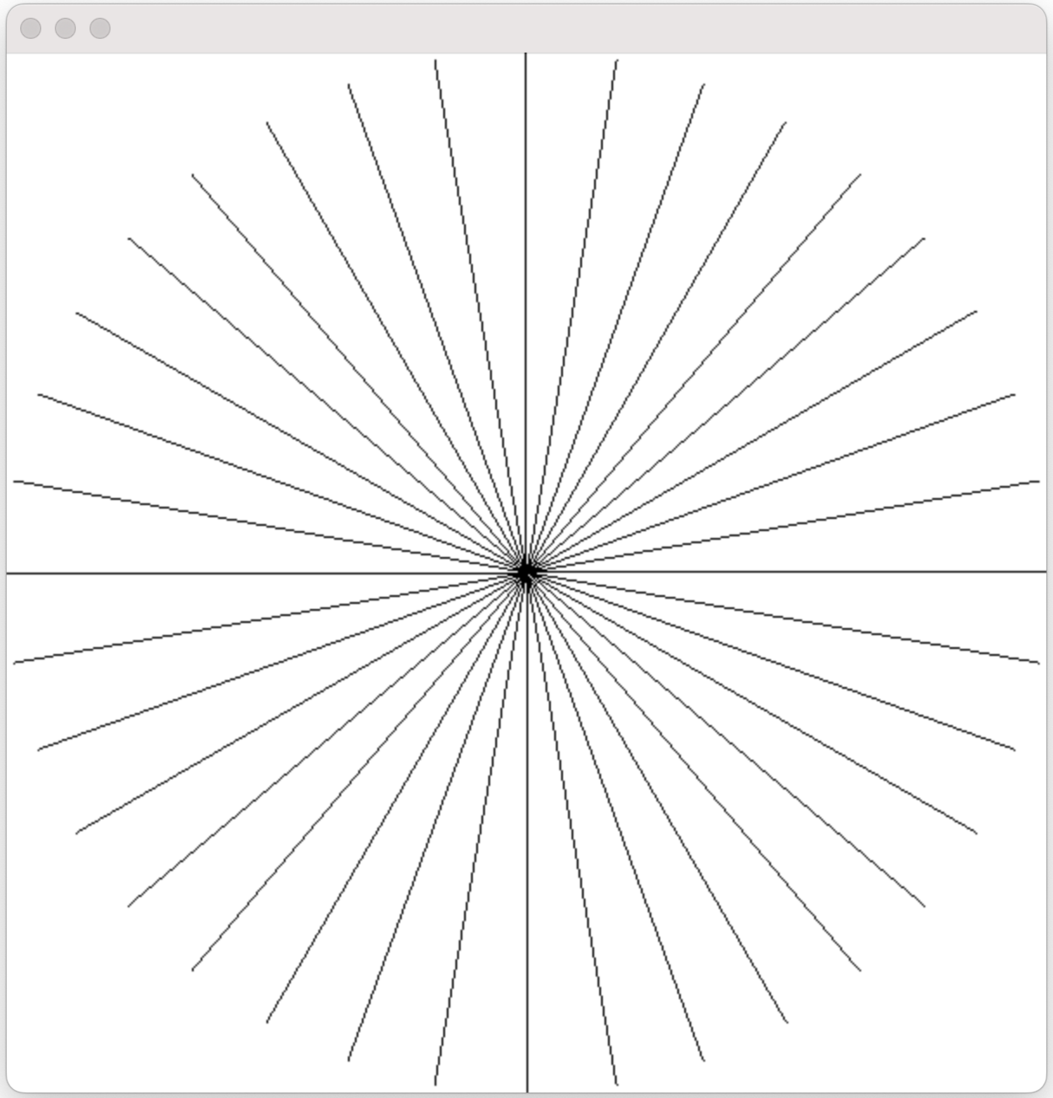
Introduktion til Programmering og Problemløsning (PoP)

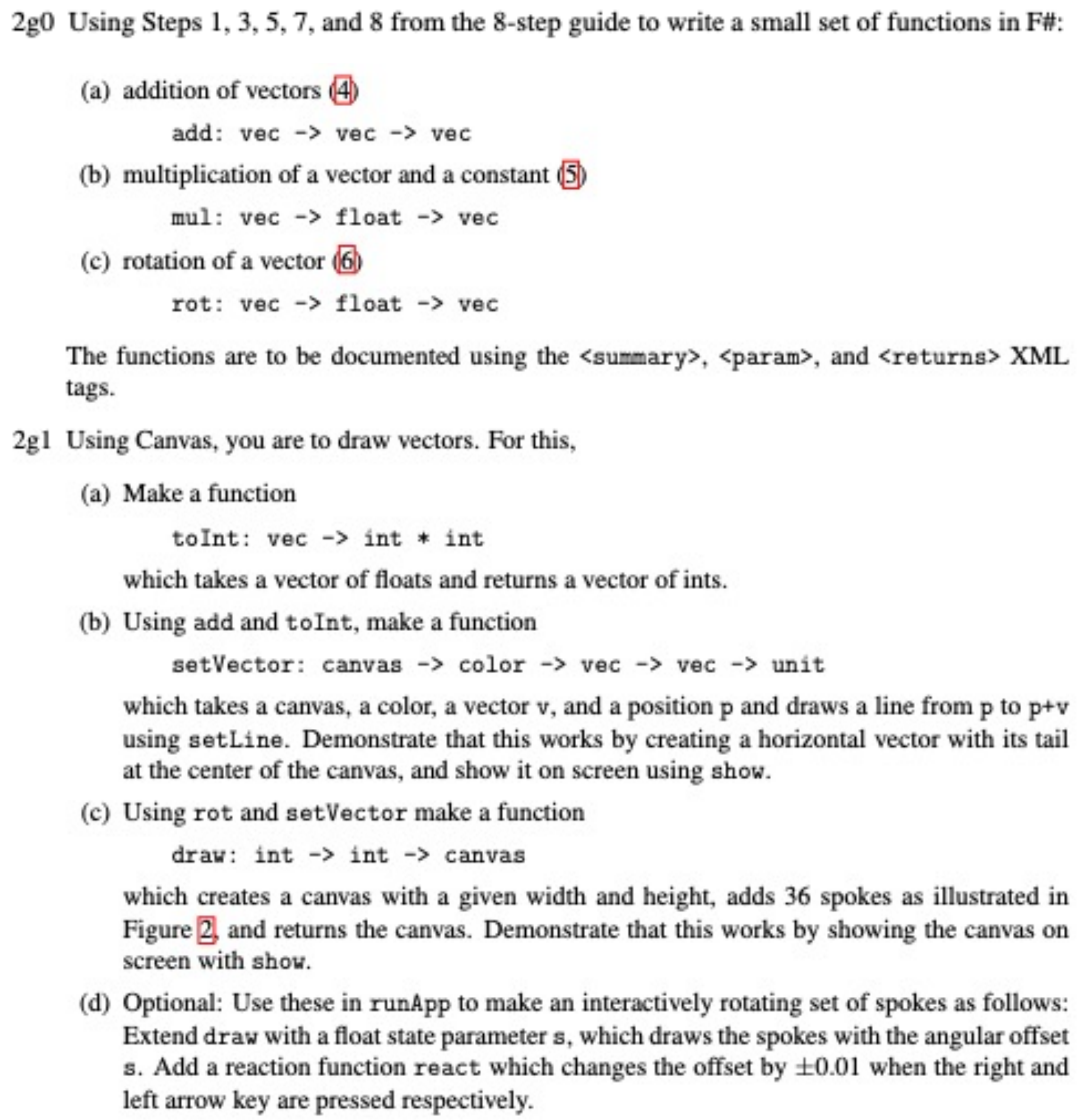
Lister

Jon Sporring
Department of Computer Science
2021/10/07

UNIVERSITY OF COPENHAGEN

Gennemgang af afleveringsopgave 2g





2g0 Using Steps 1, 3, 5, 7, and 8 from the 8-step guide to write a small set of functions in F#:

(a) addition of vectors (4)

```
add: vec -> vec -> vec
```

(b) multiplication of a vector and a constant (5)

```
mul: vec -> float -> vec
```

(c) rotation of a vector (6)

```
rot: vec -> float -> vec
```

The functions are to be documented using the `<summary>`, `<param>`, and `<returns>` XML tags.

2g1 Using Canvas, you are to draw vectors. For this,

(a) Make a function

```
toInt: vec -> int * int
```

which takes a vector of floats and returns a vector of ints.

(b) Using add and `toInt`, make a function

```
setVector: canvas -> color -> vec -> vec -> unit
```

which takes a canvas, a color, a vector v, and a position p and draws a line from p to p+v using `setLine`. Demonstrate that this works by creating a horizontal vector with its tail at the center of the canvas, and show it on screen using `show`.

(c) Using `rot` and `setVector` make a function

```
draw: int -> int -> canvas
```

which creates a canvas with a given width and height, adds 36 spokes as illustrated in Figure 2, and returns the canvas. Demonstrate that this works by showing the canvas on screen with `show`.

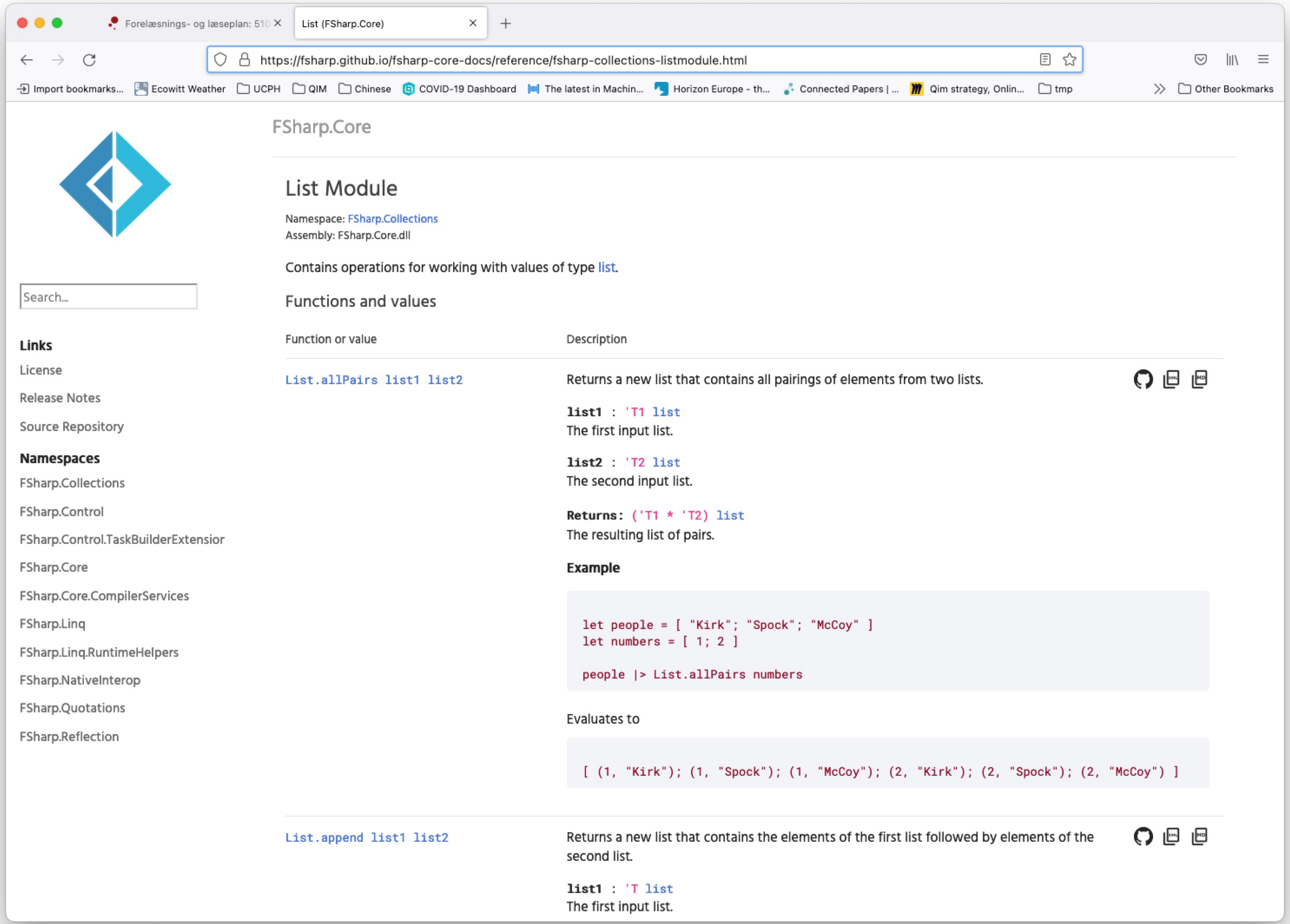
(d) Optional: Use these in `runApp` to make an interactively rotating set of spokes as follows: Extend `draw` with a float state parameter `s`, which draws the spokes with the angular offset `s`. Add a reaction function `react` which changes the offset by ± 0.01 when the right and left arrow key are pressed respectively.

Hvad er længden af lst m.m.

https://tinyurl.com/2etvjamj

Listemodulet:

https://fsharp.github.io/fsharp-core-docs/reference/fsharp-collections-listmodule.html



FSharp.Core

List Module

Namespace: FSharp.Collections
Assembly: FSharp.Core.dll

Contains operations for working with values of type list.

Functions and values

Function or value	Description
`List.allPairs list1 list2`	Returns a new list that contains all pairings of elements from two lists.

list1 : 'T1 list
The first input list.

list2 : 'T2 list
The second input list.

Returns: ('T1 * 'T2) list
The resulting list of pairs.

Example

```
let people = [ "Kirk"; "Spock"; "McCoy" ]
let numbers = [ 1; 2 ]

people |> List.allPairs numbers
```

Evaluates to

```
[ (1, "Kirk"); (1, "Spock"); (1, "McCoy"); (2, "Kirk"); (2, "Spock"); (2, "McCoy") ]
```

Function or value	Description
`List.append list1 list2`	Returns a new list that contains the elements of the first list followed by elements of the second list.

list1 : 'T list
The first input list.

List.fold og List.foldBack

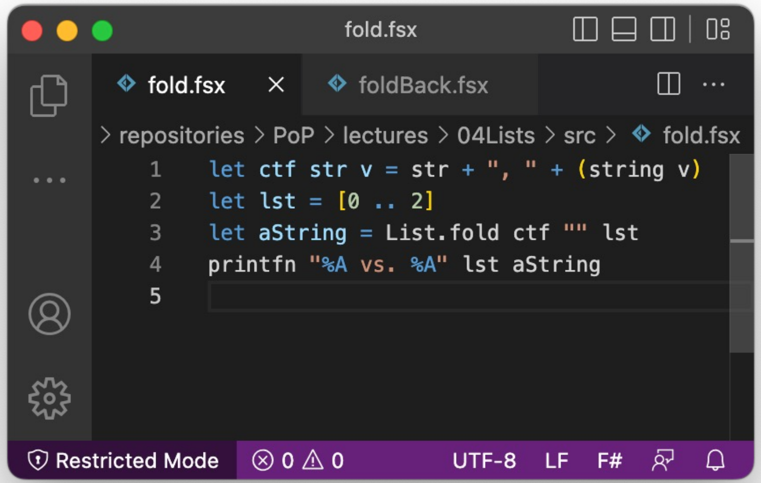
Hvad er typen?

```
% dotnet fsi

Microsoft (R) F# Interactive version 12.0.4.0 for F# 6.0
Copyright (c) Microsoft Corporation. All Rights Reserved.

For help type #help;;

> List.fold;;
val it: (('a -> 'b -> 'a) -> 'a -> 'b list -> 'a)

> List.foldBack;;
~~val it: (('a -> 'b -> 'b) -> 'a list -> 'b -> 'b)~~
val it: (('b -> 'a -> 'a) -> 'b list -> 'a -> 'a)
```

Vilkårlig type 'a og 'b

List.fold og List.foldBack

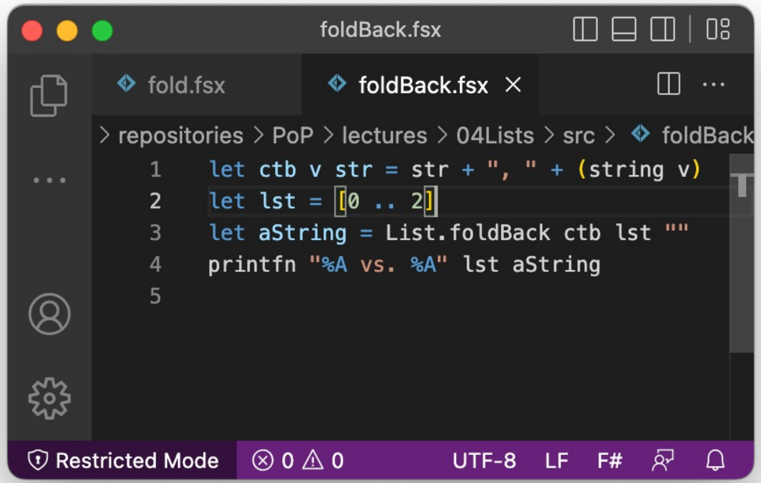
List.fold



```
let ctf str v = str + ", " + (string v)
let lst = [0 .. 2]
let aString = List.fold ctf "" lst
printfn "%A vs. %A" lst aString
```

List.fold ctf "" [0 .. 2]
~> ctf (ctf (ctf "" 0) 1) 2
~> ctf (ctf ("" + ", " + "0") 1) 2
~> ctf (ctf ", 0" 1) 2
~> ctf (", 0" + ", " + "1") 2
~> ctf ", 0, 1" 2
~> ", 0, 1" + ", " + "2"
~> ", 0, 1, 2"

List.foldBack

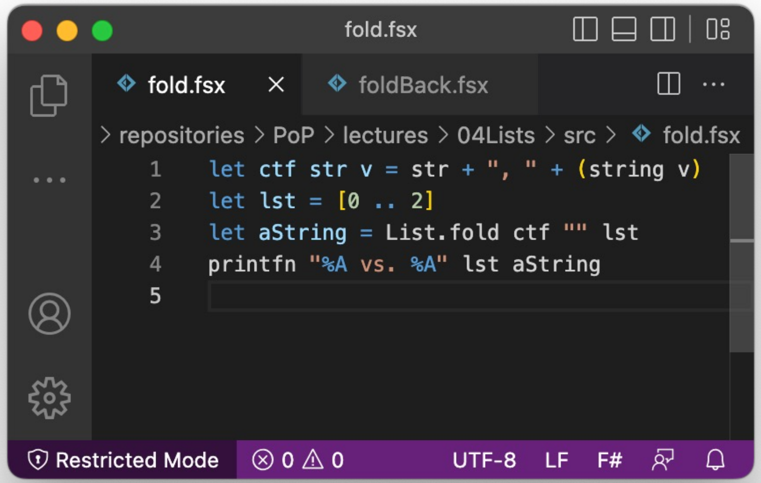


```
let ctb v str = str + ", " + (string v)
let lst = [0 .. 2]
let aString = List.foldBack ctb lst ""
printfn "%A vs. %A" lst aString
```

List.foldBack ctf "" [0 .. 2]
~> ctb 0 (ctb 1 (ctb 2 ""))
~> ctb 0 (ctb 1 ("" + ", " + "2"))
~> ctb 0 (ctb 1 ", 2")
~> ctb 0 (", 2" +", " + "1")
~> ctb 0 ", 2, 1"
~> ", 2, 1" + ", " + "0"
~> ", 2, 1, 0"

Hvordan undgår mand det foranstillede ", "?

List.fold

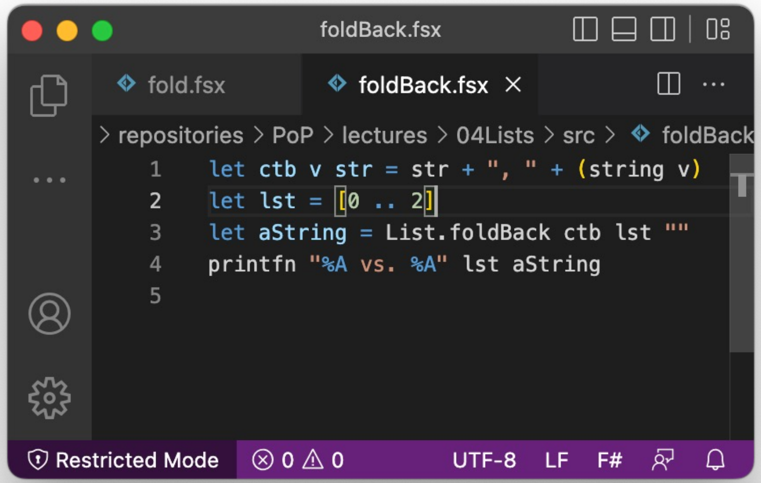


fold.fsx

> repositories > PoP > lectures > 04Lists > src > fold.fsx

```fsharp
let ctf str v = str + ", " + (string v)
let lst = [0 .. 2]
let aString = List.fold ctf "" lst
printfn "%A vs. %A" lst aString
```

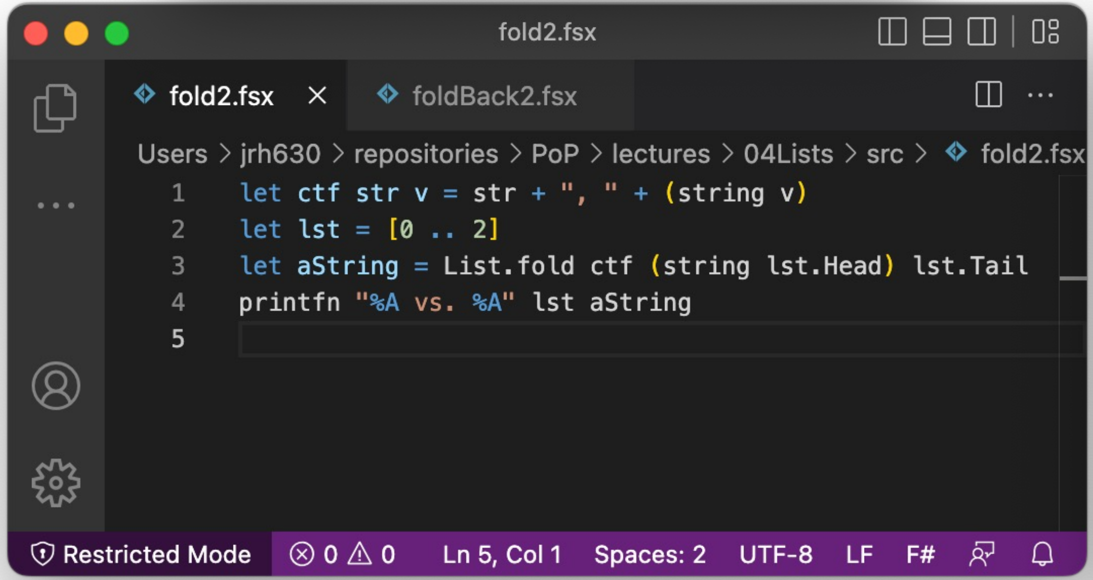
List.foldBack



foldBack.fsx

> repositories > PoP > lectures > 04Lists > src > foldBack

```fsharp
let ctb v str = str + ", " + (string v)
let lst = [0 .. 2]
let aString = List.foldBack ctb lst ""
printfn "%A vs. %A" lst aString
```

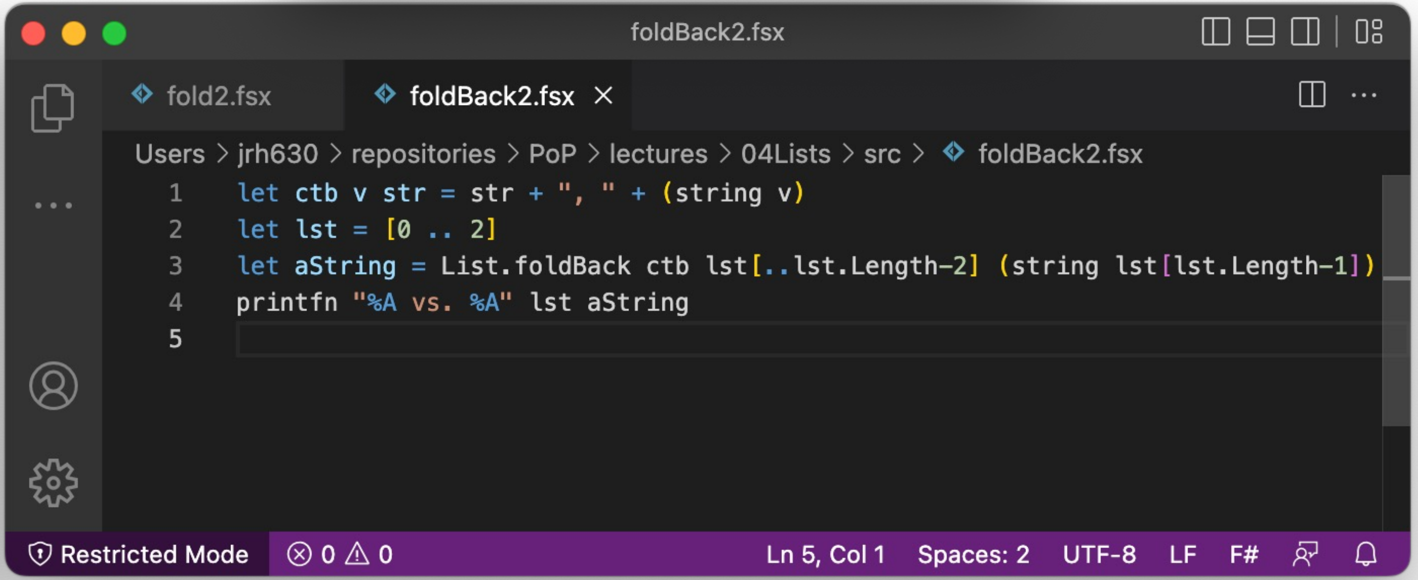


fold2.fsx

Users > jrh630 > repositories > PoP > lectures > 04Lists > src > fold2.fsx

```fsharp
let ctf str v = str + ", " + (string v)
let lst = [0 .. 2]
let aString = List.fold ctf (string lst.Head) lst.Tail
printfn "%A vs. %A" lst aString
```

```
% dotnet fsi fold2.fsx
[0; 1; 2] vs. "0, 1, 2"
```



foldBack2.fsx

Users > jrh630 > repositories > PoP > lectures > 04Lists > src > foldBack2.fsx

```fsharp
let ctb v str = str + ", " + (string v)
let lst = [0 .. 2]
let aString = List.foldBack ctb lst[..lst.Length-2] (string lst[lst.Length-1])
printfn "%A vs. %A" lst aString
```

```
% dotnet fsi foldBack2.fsx
[0; 1; 2] vs. "2, 1, 0"
```

List.fold og List.foldback med anonyme funktioner

Sum en liste af heltal:

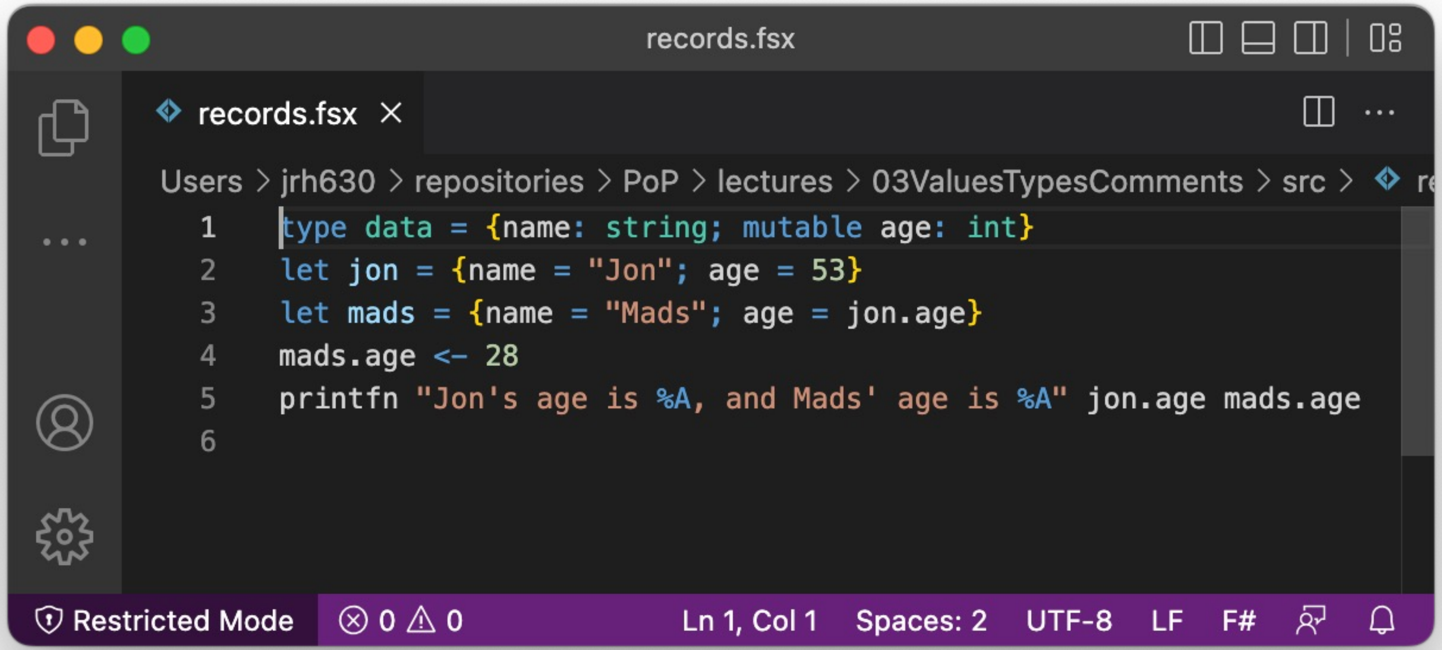
```
let add a b = a+b
List.fold add 0 [1..5]


List.fold (fun acc elm -> acc + elm) 0 [1..5]
List.foldBack (fun elm acc -> acc + elm) [1..5] 0
```

Sekvens af udtryk

Debugging med printfn og sekvenser af udsagn:

```
List.fold (fun acc elm -> printfn "(%d,%d)" elm acc; acc + elm) 0 [1..5]
List.foldBack (fun elm acc -> printfn "(%d,%d)" elm acc; acc + elm) [1..5] 0
```

Hjernevrider uge 3


records.fsx
records.fsx
Users > jrh630 > repositories > PoP > lectures > 03ValuesTypesComments > src > re
Restricted Mode
0 0
Ln 1, Col 1
Spaces: 2
UTF-8
LF
F#

```
type data = {name: string; mutable age: int}
let jon = {name = "Jon"; age = 53}
let mads = {name = "Mads"; age = jon.age}
mads.age <- 28
printfn "Jon's age is %A, and Mads' age is %A" jon.age mads.age
```

- ○ Jon's age is 53, and Mads' age is 28
- ○ Jon's age is 28, and Mads' age is 28
- ○ Ingenting, der er en fejl
- ○ Ingen af delene
- ○ Ved ikke

Spørgsmål