D2 前端技术论坛
D2 FRONTEND TECHNOLOGY FORUM
精心

第十六届D2前端技术论坛

ReScript: the sound typescript that scales

Hongbo Zhang

About me: working in PL since 2009

- Author of ReScript(rescript-lang.org)
- Ex-core contributor to OCaml and Camlp4
- Author of Fan meta programming thesis
- Github: @bobzhang
- Twitter: @bobzhang1988
- Zhihu: people/hongbo_zhang

Outline

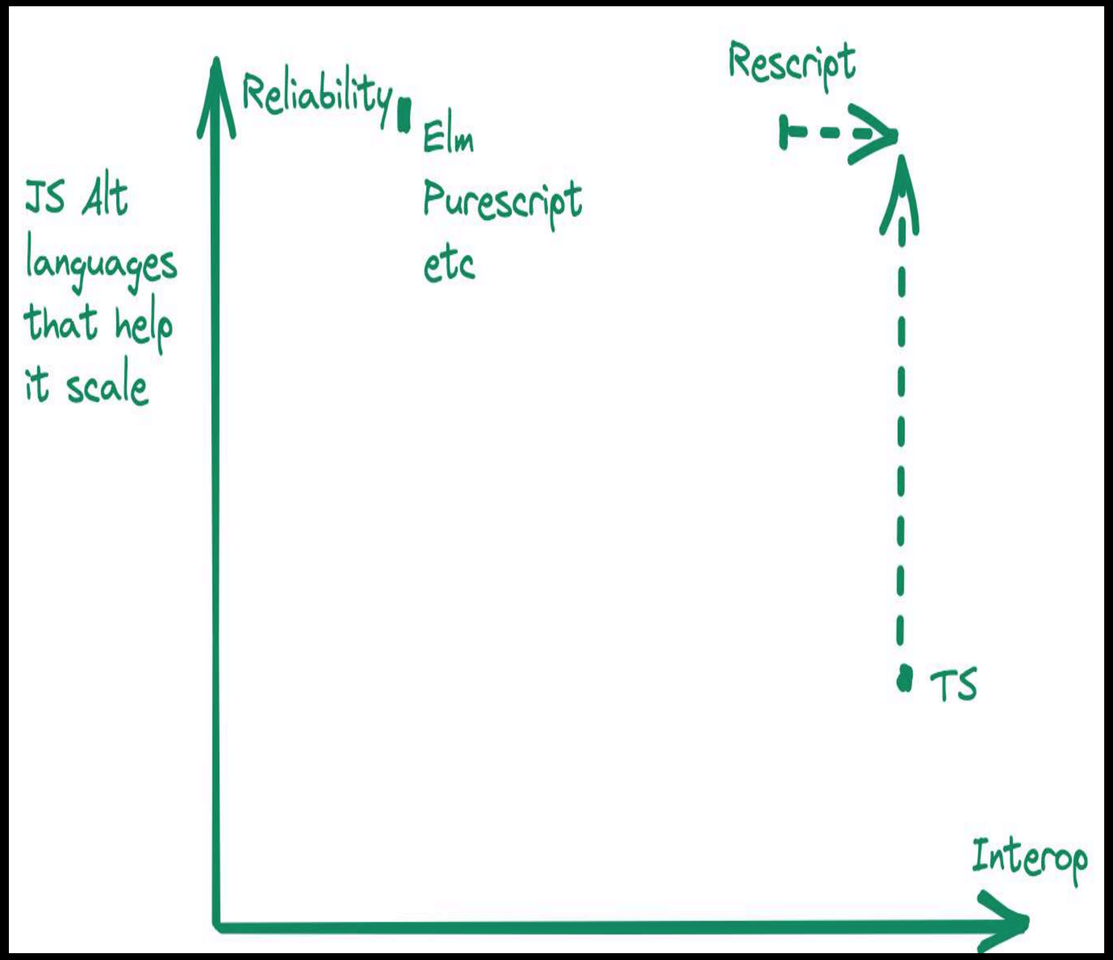
01 ReScript intro and history

02 Data oriented programming

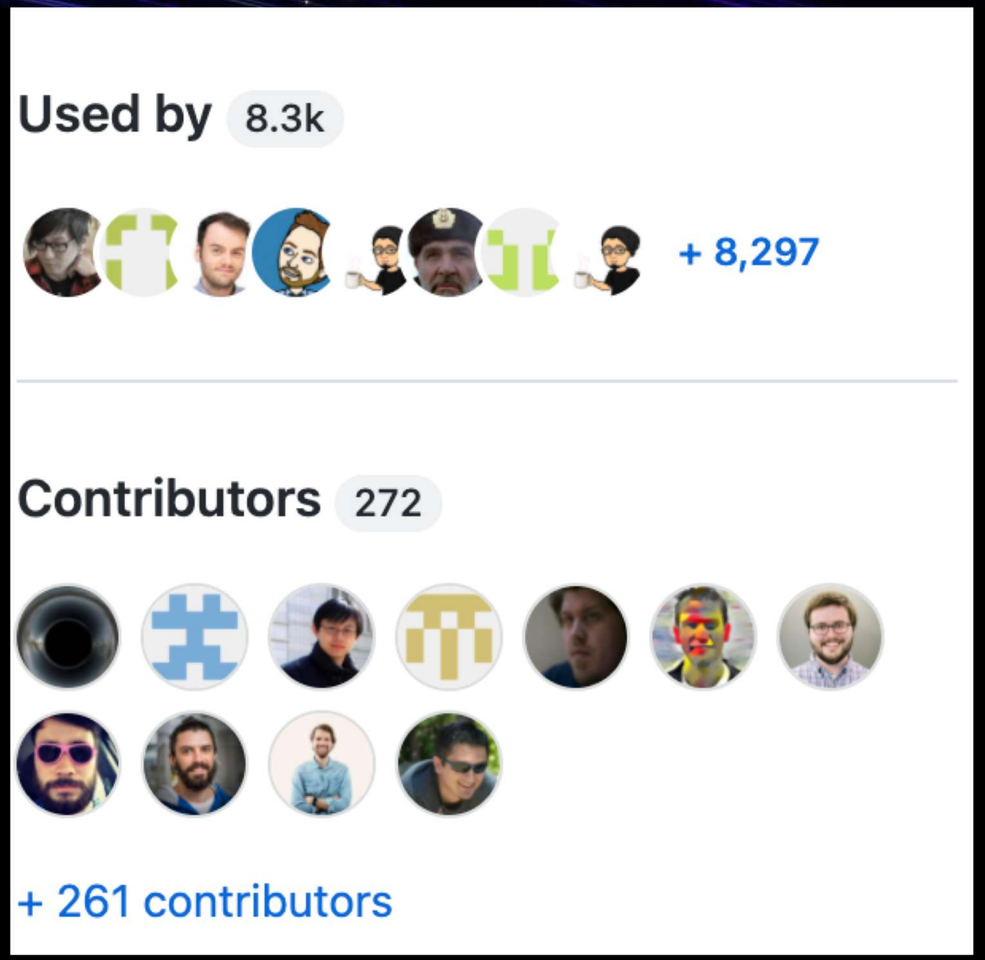
03 Example: red black tree

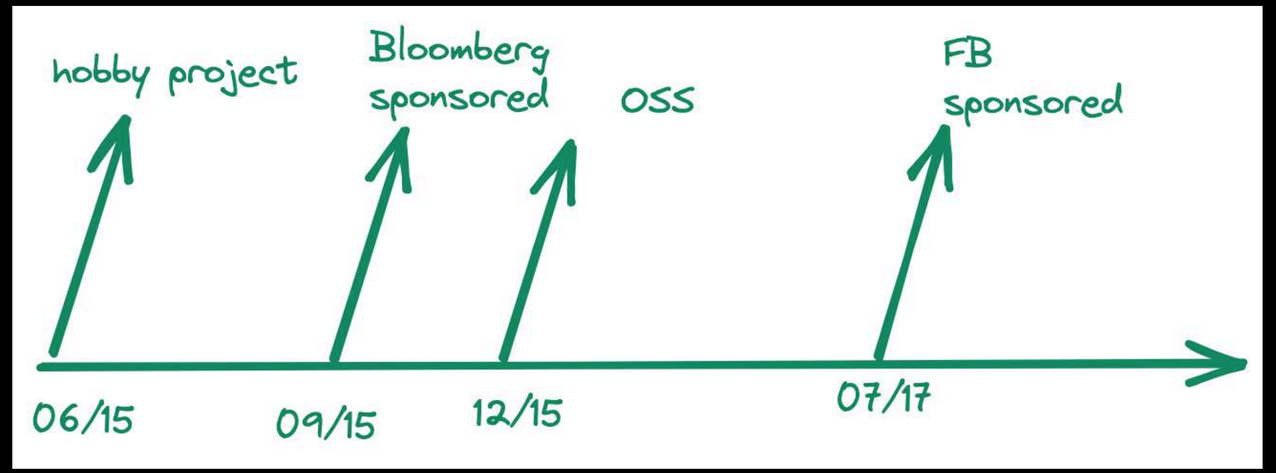
Rescript: a statically typed functional language

- Optimized for JS platform
- Academic roots from ML
- Focus on the quality of implementations

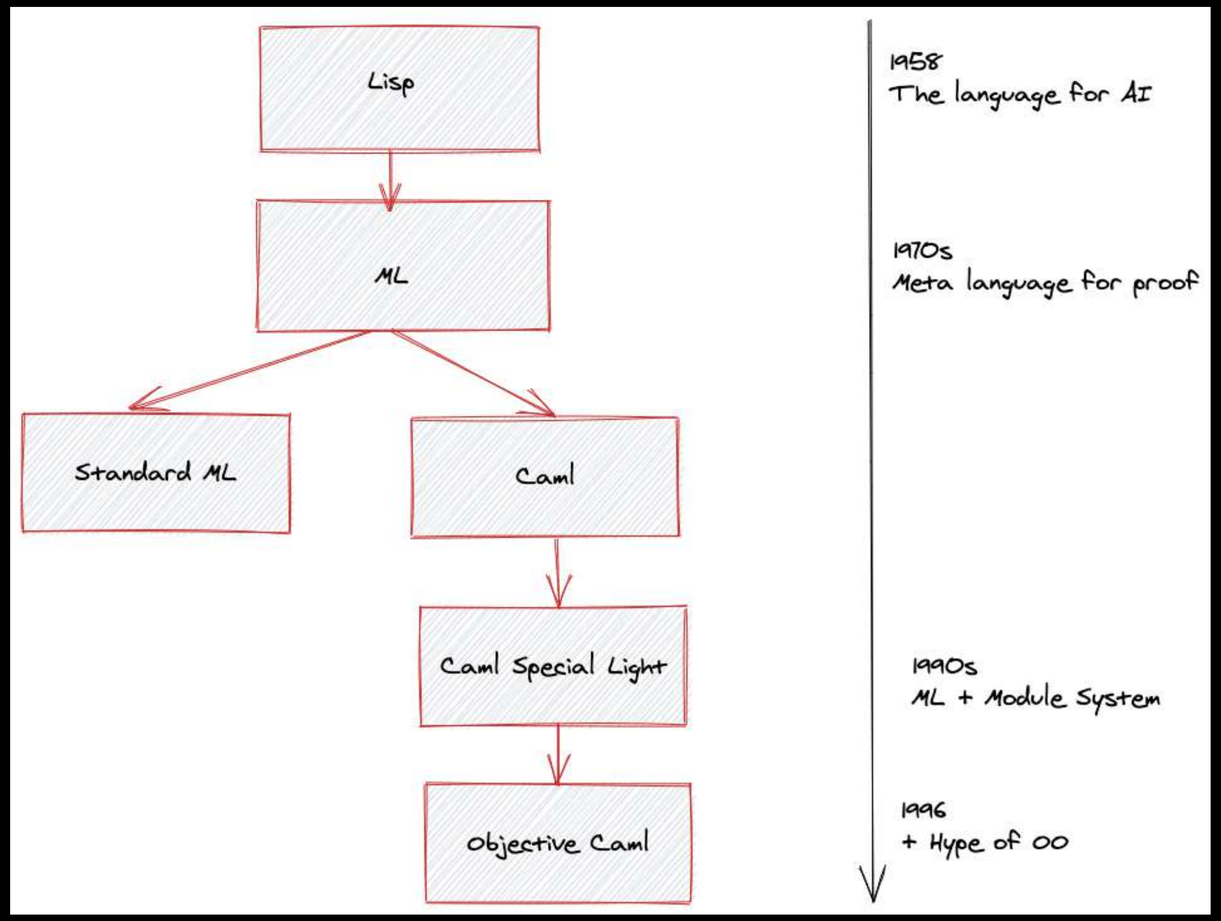


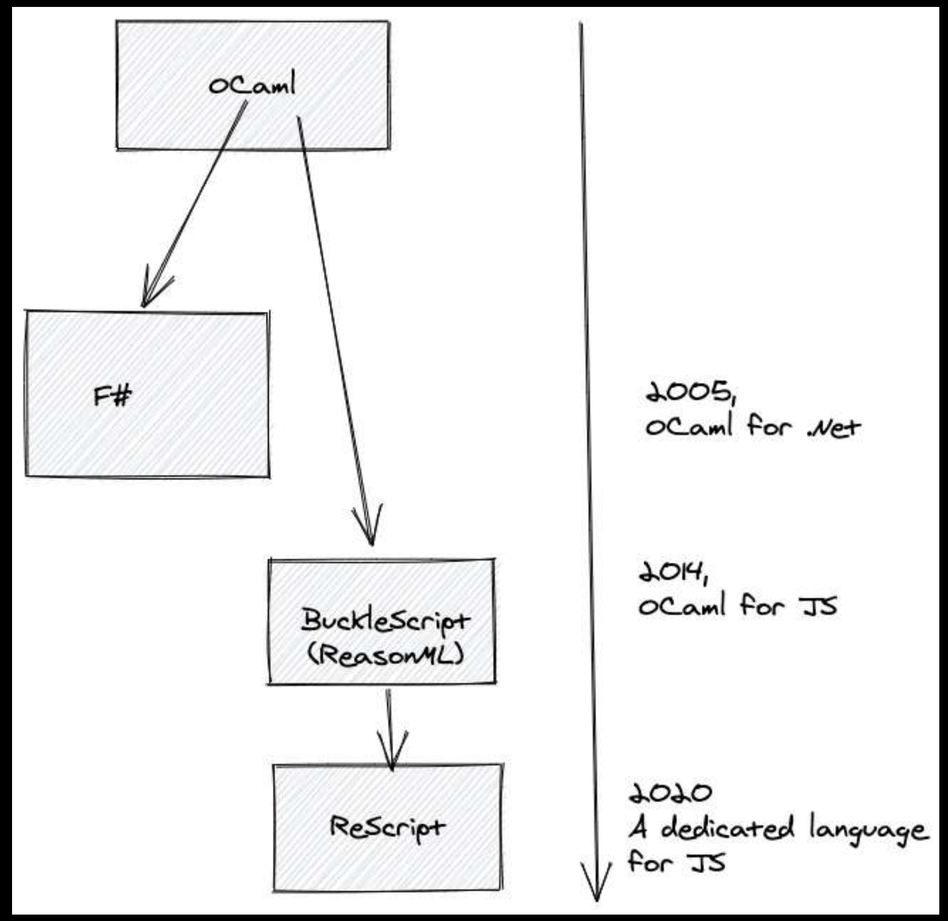
Make a programming language is easy.
Make a language that is trusted by other companies is very difficult





The history of ReScript





International impact of Rescript

第一个真正意义上中国人制作的具有海外影响力的通用编程语言

International impact for non-English speaking countries

- ReScript in Korean

  ReScript in Korean (green-labs.github.io)

- ReScript in Spanish (Training course in Brazil)

  ES2077 | Aprenda programação funcional com React e ReScript (es2077-site.vercel.app)

- ReScript in French

  Pourquoi et comment ReScript a été utilisé pour repenser le Design System chez Arte | by Linkvalue | Apr, 2021 | Blog Linkvalue—société de conseil IT (link-value.fr)

- ReScript in Japanese

  ReScriptを学ぶ—JavaScriptを書くためのより安全な方法-パート1 (ichi.pro)

- ReScript in Germany

  ReScript – the language after TypeScript? - codecentric AG Blog

The flavor of ReScript

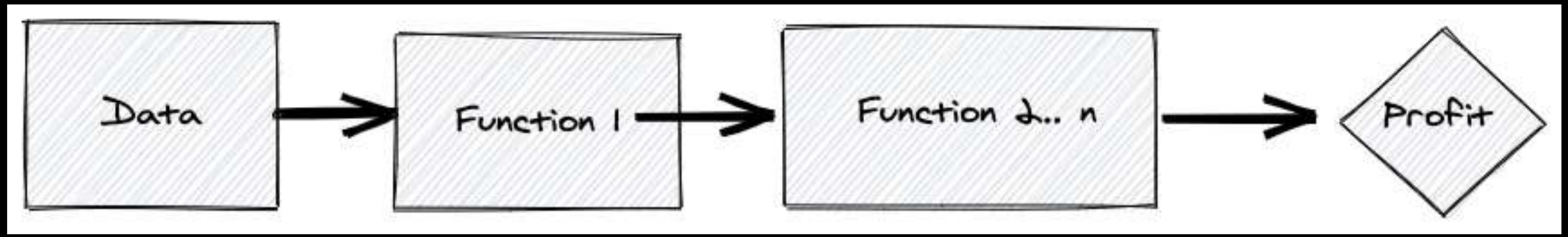
Format | Copy Share Link

```
let sum = (~a,~b)=>{
  a + b
}


sum(~a=1,~b=3)->Js.log

(~a: int, ~b: int) => int
sum(~b=2,~a=1)->Js.log
```

JavaScript | Problems | Settings

```
// Generated by ReScript, PLEASE EDIT WITH
CARE
'use strict';


function sum(a, b) {
  return a + b | 0;
}

console.log(4);

console.log(3);

exports.sum = sum;
/*  Not a pure module */
```

Data oriented programming in ReScript



Data oriented programming in ReScript

Format | Copy Share Link

```
1 "hello world" -> Js.log
```

JavaScript | Problems | Set

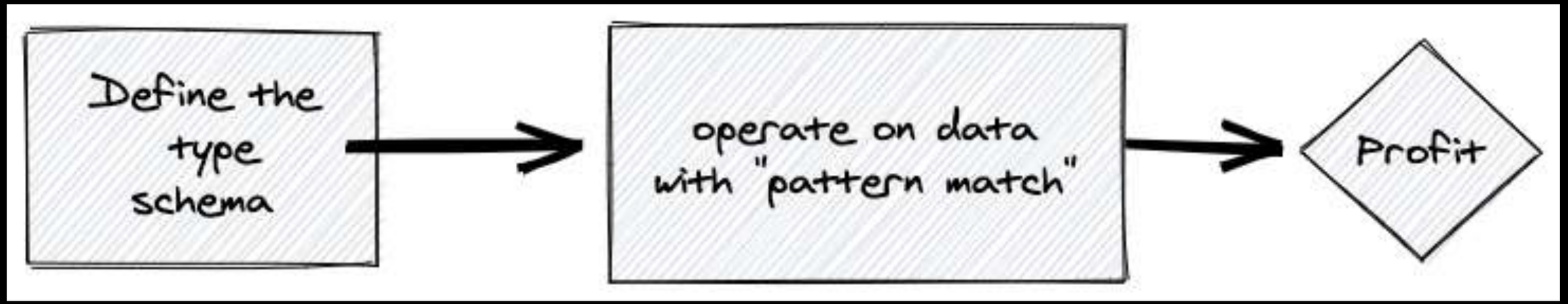
```
// Generated by ReScript, PLEASE EDIT
CARE
'use strict';


console.log("hello world");

/*  Not a pure module */
```

Data oriented programming in ReScript

Format | Copy Share Link

```
open Belt.Array
[1,2,3]
-> map(x=>x+1)
-> keep(x=> x > 2)
-> reduce((a,b)=>a+b)
-> Js.log
```

Data oriented programming in ReScript



Data oriented programming in ReScript


```
type maybeInt =
    | Some (int)
    | None

let f = (x)=>{
  switch x {
    | Some (x) => x
    | None => 0
  }
}
```

```
type maybeInt =
  | number
  | undefined

var f = (x : maybeInt)=>{
    if(typeof x === "undefined"){
        return x
    }
    return 0
}
```

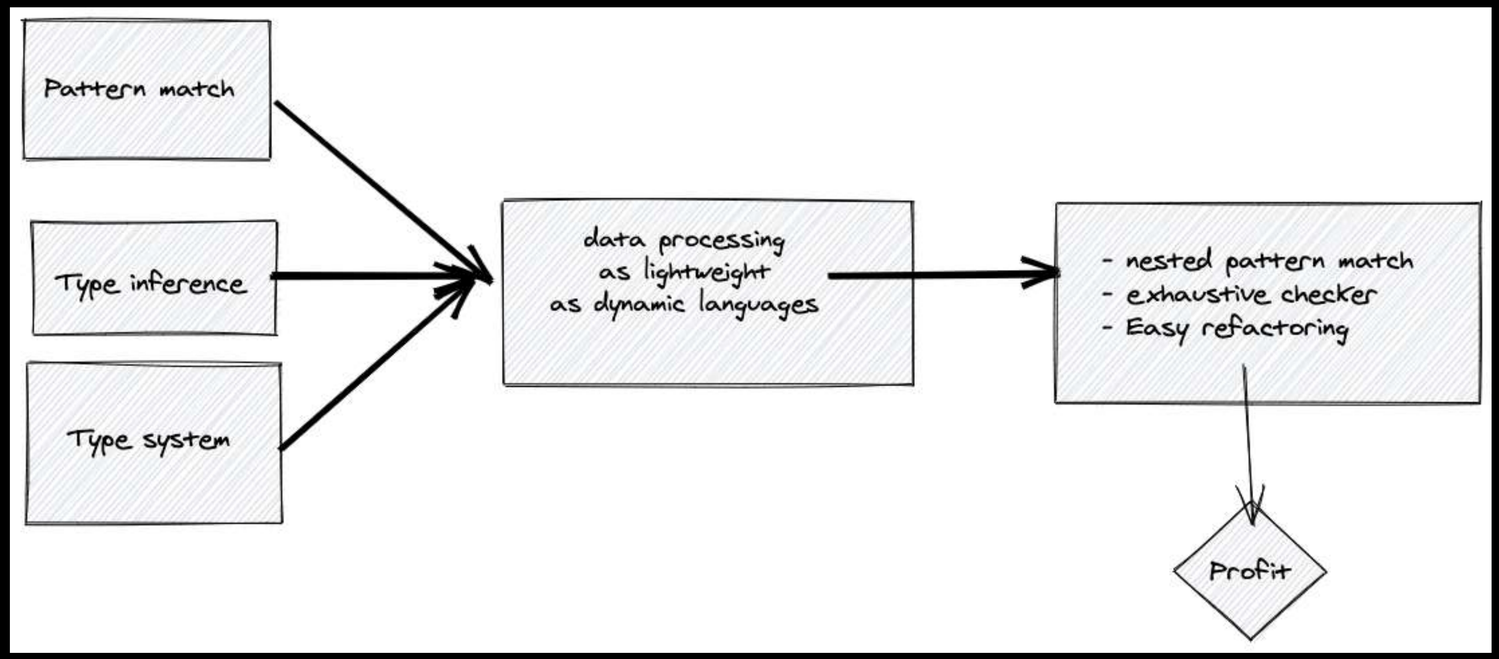
```
// Generated by ReScript, PLEASE EDIT WITH
CARE
'use strict';


function f(x) {
  if (x !== undefined) {
    return x;
  } else {
    return 0;
  }
}

exports.f = f;
/* No side effect */
```

Data oriented programming in ReScript


```
type rec list =
    | Cons (int, list)
    | Nil
let rec map = (x,f)=>{
  switch x {
    | Nil => Nil
    | Cons(x,xs)=> Cons(f(x),map(xs,f))
  }
}
```

```
type list =
    | [ number , list]
    | undefined

var map = (x : list, f : (n:number)=> number) : list => {
  if (typeof x === "undefined"){
     return undefined
  }
  var [x0,xs] = x
   return [f(x0),map(xs,f)]
}
```

Data oriented programming in ReScript



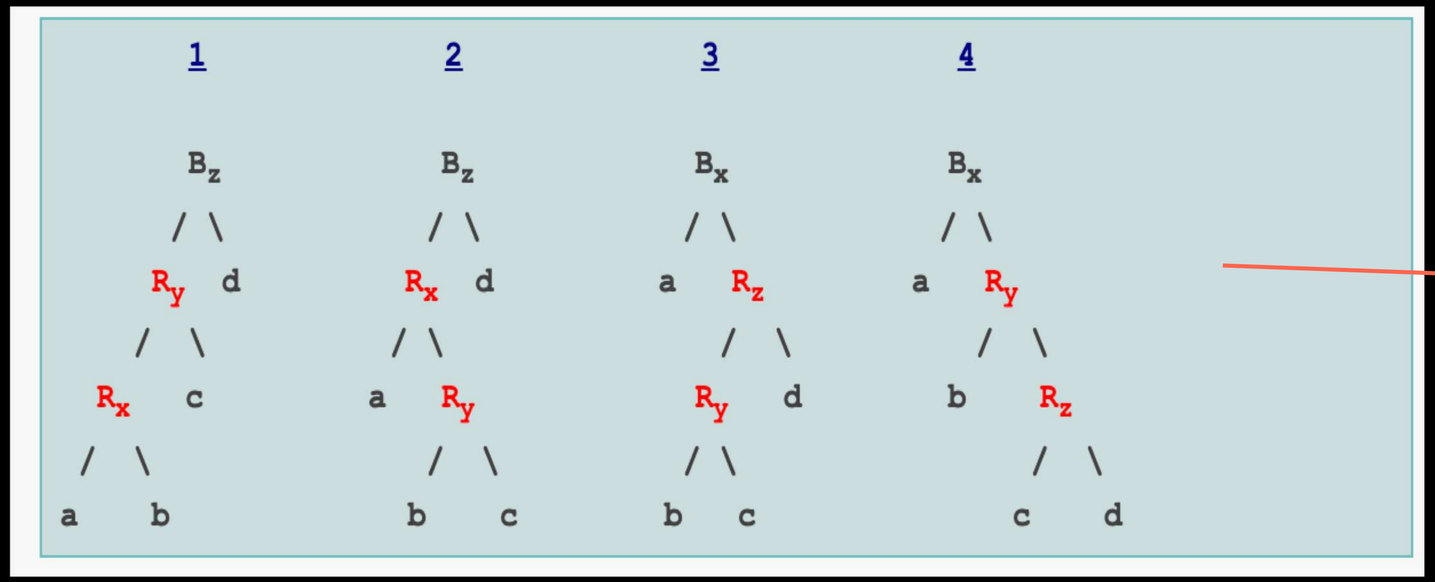
The power of pattern match

Red Black tree in ReScript: types


```
type color =
  | R
  | B

type rec tree =
  | Leaf
  | Node({val: int, l: tree, r: tree, color: color})
```

```
type color =
    | 'R'
    | 'B'

type leaf = undefined
type tree =
    | leaf
    | { val : number, l : tree , r : tree , color : color}
```

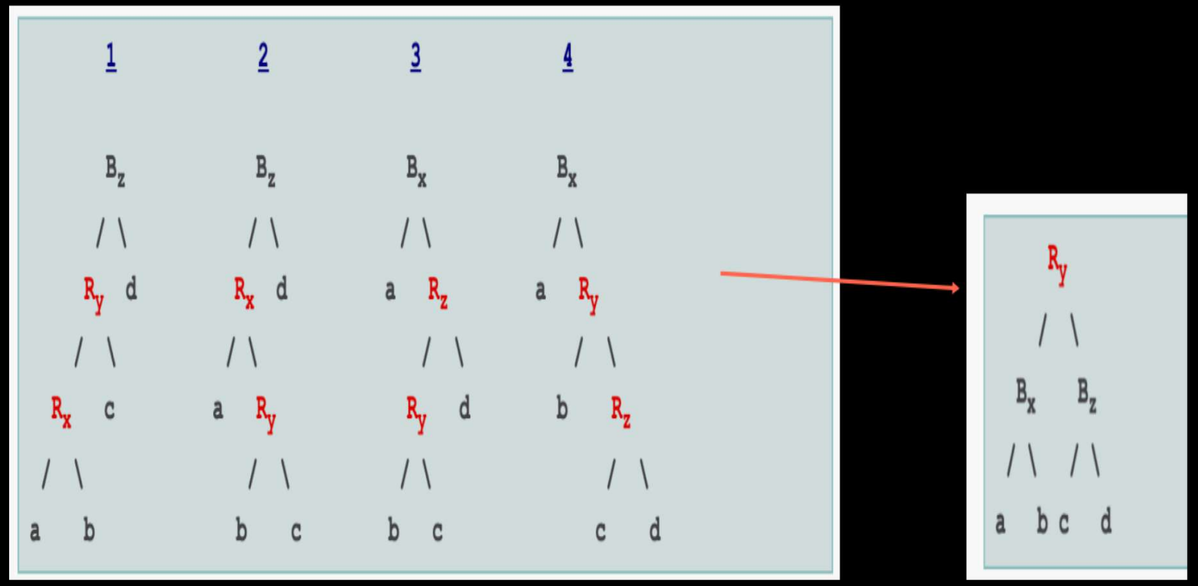
Data oriented programming in ReScript



Courtesy to Cornell cs3110

Red Black tree in ReScript: rotation


```
let rotate = tree => {
  switch tree {
  | Node({
    val: z,
    l: Node({color: R, val: y, l: Node({color: R, val: x, l: a, r: b}), r: c}),
    r: d,
  })
  | Node({
    val: z,
    l: Node({color: R, val: x, l: a, r: Node({color: R, val: y, l: b, r: c})}),
    r: d,
  })
  | Node({
    val: x,
    l: a,
    r: Node({color: R, val: z, l: Node({color: R, val: y, l: b, r: c}), r: d}),
  })
  | Node({
    val: x,
    l: a,
    r: Node({color: R, val: y, l: b, r: Node({color: R, val: z, l: c, r: d})}),
  }) =>
    Node({
      color: R,
      val: y,
      l: Node({color: B, val: x, l: a, r: b}),
      r: Node({color: B, val: z, l: c, r: d}),
    })
  | _ => tree
  }
}
```



AVL tree in ReScript

https://zhuanlan.zhihu.com/p/430867594

Thank you!

Rescript is a multi-paradigm programming language, read more in
https://rescript-lang.org/