

```
fun max(xs: int list) =  
  if null(tl xs) then hd xs  
  else  
    let  
      val x = max(tl xs)  
    in  
      if hd xs > x then hd xs  
      else x  
    end
```

从-INF开始学习Standard ML



Mike He

I am; therefore I fail.

10 人赞同了该文章

Prologue

时隔几个月终于又有时间抽出时间写一写文章了...申请季打得我措手不及，但是还好，今天早上收到了Computer Science, College of Science@Purdue University的Offer，所以喘口气歇一歇，写写文章好了...

这个系列的文章我打算将关注点放在一个函数式编程的语言——Standard ML——上。

最近闲来无事在某MOOC上学习程序设计语言这个课程，感觉这个ML还是蛮有意思的...如某MOOC的Professor所讲，希望dalao们看这系列文章时`get rid of other languages you've learned`，将其作为一个全新的语言看待。

本篇文章将关注：

- Data Binding
- Shadowing
- Functions
- Expressions

▲ 赞同 10 ▼

● 6 条评论



这里所说的Data Binding其实就是变量赋值(绑定)。SML的数据绑定语法为：

```
val x = <value> : <type>
```

其中若不加<value> : <type>, ML会进行类型推断。

那么数据绑定的过程中, 实际上发生了什么? SML将变量信息存储在两个**环境**(Environment)中:

- 静态环境(Static Environment): 存储数据与字面量的类型绑定的信息
- 动态环境(Dynamic Environment): 存储数据与字面量绑定的信息

E.g:

```
val x = "6662333" : string
```

这时, 在静态环境中: $x \rightarrow \text{string}$; 在动态环境中: $x \rightarrow "6662333"$ 。That's it, very simple.

Shadowing

在数据绑定之后是不能进行修改的。如果在SML中尝试使用 $\text{<identifier> = <value>}$, 你会发现REPL会返回一个异常或者 **bool** 类型的值。因为SML的 *equality assertion* 就是 = 符号...所以, 修改数据事实上是**覆盖**原有数据, 也就是本段标题的 *Shadowing*。具体操作十分Simple:

```
(* Static Env: x -> int *)
(* Dynamic Env: x -> 6662333 *)
val x = 6662333;
(* After Shadowing *)
(* Static Env: x -> string *)
(* Dynamic Env: x -> "6662333" *)
val x = "6662333";
```

因此这里变量的自增也是 $\text{val <identifier> = <identifier> + offset;}$

Functions

▲ 赞同 10



● 6 条评论



```
fun <function_name>(v1: t1, v2: t2) = <function_body>
```

其中 $v_1, v_2 \dots v_n$ 是形参的*identifier*。 $t_1, t_2 \dots t_n$ 是每个形参所对类型。 $\langle function_body \rangle$ 可以写在一行(Sugar for 压行选手--手动划掉)也可以换行写, 并且 $\langle function_body \rangle$ 是一个能够返回计算结果的语句(不是顺序执行的代码)。例如一个 $x \rightarrow x^{**}3$ 的函数就可以这样写了:

```
fun cube(x: int) = x * x * x
```

换行写是什么样子呢? ? 简单一点, 例如判断一个数字是否是偶数:

```
fun test(x: int) =  
    if x mod 2 = 0 then true  
    else false
```

调用时: $\langle function_name \rangle(v_1, v_2, \dots, v_n)$ 即可。若该函数形参只有一个就可以不写括号, 简化为: $\langle function_name \rangle v_1$ (如`test 100`)。

Expressions

if ... then ... else! 英文写作课开课啦!

除了刚刚提到的“赋值语句”还有一个`if then else`...Syntax:

```
if e1 then v1 else v2;
```

这里的 $e1$ 是一个 **bool** 类型或者能够返回 **bool** 类型值的语句与其他编程语言不太相同(我知道和某蛇相似但是为了obey the rule, 这里就划掉吧), 这里的 $v1$ 与 $v2$ 并不是语句块, 而是一个值或者返回值的语句或者`let ... in ... end`。因此可以将其返回的值绑定在一个变量上:

```
val res = if e1 then v1 else v2;
```

当然, 判断语句是可以连续的:

```
val res = if e1 then v1  
          else if e2 then v2
```

▲ 赞同 10

▼ 6 条评论



let ... in ... end~啊，进来啊~~

刚刚看到Functions之后就有人会问，既然一个函数里只能包含一个返回结果的语句或语句块，如果我想要引进其他变量怎么办呢？这时就要请来这个(名字在有点h的title里)语句`let ... in ... end`。你可以在`let ... in`之间加入任何你想要绑定的数据，甚至定义一个局部函数(*local function*)。然后你能且仅能在`let ... in`和`in ... end`之间使用之前绑定好的数据。**这里要注意，所有的数据都需要在绑定之后再使用**

作用域

`let ... in ... end`的作用域：在`let ... in`之间可以访问到其所在作用域的所有数据。如果在这个部分重新绑定函数形参或者其父`let ... in ... end`语句块中已绑定好的变量，被操作的变量会由于**Shadowing**而失去之前的值。

`let ... in`中绑定的数据只能被绑定后在`let ... in`中被访问；可以在`in ... end`中被使用。

最后附上运算操作符

算术运算

加法：+ 减法：- 乘法：* 除法(int)：div 除法(real)：/ 取模：mod

相反数(负号)：~

逻辑与比较

或者：orelse 并且：andalso 非：not

大于：> 小于：< 等于：= 不等于：<>

(剩下以此类推啦....)

最后的最后

▲ 赞同 10 ▼

● 6 条评论



编辑于 2018-01-16

函数式编程 计算机科学 Standard ML

文章被以下专栏收录



编程杂谈
记录编程的日常 & 学习的过程 & 奇技淫巧

进入专栏

推荐阅读



Monad via LINQ (0)

Eli Chen

Java 8 Stream API 中 map 和 flatMap 方法使用详解

java 8 stream api 中有两
map和flatMap非常实用，
景也非常广泛，能极大提
率。下面我们详细介绍一
方法的用法。map方法我
示例：把一个整数列表转
陈大侠 发表于

6 条评论

切换为时间排序

写下你的评论...

😊



知乎用户
Mike深吸了一口气并打开了CSGO。。

2 年前

👍 赞

▲ 赞同 10 ▼ 💬 6 条评论 ➦

知乎

首发于
编程杂谈

应该就是Coursera上牛益帆入手的加个口味吧！

👍 赞



Mike He (作者) 回复 陈力

2 年前

被乃发现了:D

👍 赞



知乎用户

2 年前

大佬换vscode了orz

👍 赞



任凭

6 个月前

你好，我想问一下要怎么在vscode上使用standard ML，我安装standard ML扩展以后，编辑.sml文件时，右下角会显示standard ML，但是我右键run code，就会提示“Code language not supported or defined.”我在google上找，但是讨论比较少，请问您当时遇到了这个问题吗？

👍 赞



Mike He (作者) 回复 任凭

6 个月前

我一般直接用terminal运行，不是很清楚vscode的支持。

👍 赞

▲ 赞同 10 ▼

💬 6 条评论

