



# Language Update

## *Standard MLer* から見た *ML* 概要

上野雄大

(北陸先端科学技術大学院大学)

katsuu@jaist.ac.jp

検閲済





# What is ML?



# What is ML?

---



~~ML  
=  
Mailing List~~



# What is ML?

---



**ML**

**||**

**Markup Language**



# What is ML?

---

**ML**

**||**

**Meta Language**

# What is ML?

---

ややこしい...

なまえ  
じゅーよー

# What is ML?

---

- 関数型言語
- pure じゃない
- 正格
- 強い型付け
  - 型推論：多相型型推論は ML が元祖
- 型理論をベースとした強力なモジュール機構
  - structure, signature, functor

# What is ML?

---

- 2つの方言
  - Standard ML
  - Caml

# Standard ML

- 本家
- 厳密な言語定義
  - “Specification”ではなく“Definition”
  - 式・型の定義と89の型付け規則、97の評価規則
- いろいろな人のいろいろな実装
  - SML/NJが最も完成度が高い

# Objective Caml

---



- フランス方言
- 仕様 = 実装
- INRIA が公開する実装のみ



# 違い

- Standard ML
  - 洗練された言語仕様
  - 覚えることや驚きが少ない
  - でも機能も少ない
- Objective Caml
  - 研究 実装のサイクルが速い
  - 機能いっぱい
  - 「使える」実装が揃っている

互換性なし

# MLの特徴



---

## 世の中は汚れている

- Haskellはmonadとかで言語のpurityを守った
- MLは言語のpurityを守ることを諦めた



# MLの特徴

俺ぁ汚れちゃったよ...

- いいこと
  - 例外機構
  - 直感的な I/O 処理
  - ユーザーに難しい概念を無理強いしない
- わるいこと
  - Value Polymorphism
    - ：副作用のある言語の型付けの理論的限界

# MLの特徴

- 読みやすい構文
  - 意味不明な記号の羅列にならない
  - 括弧だらけにならない
  - parser が簡単に書ける
- 高速な実行
  - gcc に迫る性能 (OCaml)
  - 停止性では遅延評価な言語に負ける

# MLの特徴

- 関数型プログラミング
  - 「どうするか (how)」より「何をするか (what)」を素朴に書く
  - 大きな処理は小さな処理の組み合わせ
- 強い型付けによるテストの軽減
  - 型 OK = core を吐かないことの証明

# MLの特徴

---

コンパイラー発で  
動く快感

# MLの特徴



---

- 正格 ... 書いた通りに実行
  - きっと直感的
  - コードから処理を追える = デバッグが楽





関数型言語の  
くせに  
printfデバッグが  
できる



MLにprintfはありません

# MLの特徴



---

型チェックが通るようになるまでは  
茨の道

(ぼそ



# コード例 (quick sort)

```
local
  fun less x l = List.filter (fn i => i <= x) l
  fun greater x l = List.filter (fn i => i > x) l
in
  fun qsort [] = []
    | qsort (x::xs) =
      qsort (less x xs)
      @ [x]
      @ qsort (greater x xs)
end
```

# コード例 (素数列挙 1)

```
local
  fun nextprime n primes =
    if List.exists (fn x => n mod x = 0) primes
    then nextprime (n + 1) primes
    else n :: primes
in
  fun prime [] = [2]
    | prime (x::xs) =
      nextprime (x + 1) (x::xs)
end
```

# コード例 (素数列挙2)

```
val prime = (2, fn x => x mod 2 = 0)
fun nextprime (n, f) =
  if f n then prime (n + 1, f)
  else (n, fn x => x mod n = 0 orelse f x)
```

正格な言語は無限個の列挙は苦手



# Language Update



# 歴史



1970年代 : ML 誕生

1980年代 : 理論的な整備が進む

1985年 : Caml 誕生

1990年 : SML '90

1997年 : SML '97

- 言語の定義は97年から変化していない (SML)
- 研究は活発に行われている



# Implementation Update

- SML/NJ
  - stable: 110.0.7 (2000-09-28)
  - unstable: 110.55 (2005-07-20)
  - 言語自体は97年から変化無し
  - 周辺環境は激変
- Objective Caml
  - 3.08.4 (2005-08-16)
  - 新機能がホイホイ入る

# Language Update

---

最近 : F#

最近 : 美しい日本のMLコンパイラ

未来 : next generation of ML

# F#



---

- MSR に移った ML 研究者が関与
- OCaml ベース
- .NET Platform
- 他言語 (C# とか) との interoperability が高いらしい
- .NET のサポートによる実用的な機能の充実



# 美しい日本のMLコンパイラ

- 2004年度第1回IPA未踏に採択
- MinCaml
- 教育用トイコンパイラ
- MLのエッセンスだけを取り出せば、たった2000行程度で効率の良いコンパイラが作れてしまうことの実証

# Next generation of ML



- Polymorphic record & variant
- Rank-n polymorphism
- Interoperable calculus
- Register allocation by proof transformation
- Bitmap calculus
- Unboxed float

理論的には完成。実装待ち



# おわりに

---

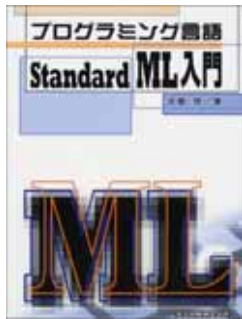
ML は

- 古くて
- 新しくて
- 汚くて
- 美しい

関数型言語です。

ML makes you a cool hacker!

# 教科書とか



プログラミング言語 Standard ML 入門  
ISBN:4320120248



プログラミング言語 ML  
ISBN:4756116418



The Definition of Standard ML (Revised)  
ISBN:0262531814

# 参考 URL

---



- Standard ML of New Jersey

- <http://www.smlnj.org/>

- Objective Caml

- <http://www.ocaml.jp/>

- <http://caml.inria.fr/>

- MinCaml

- <http://min-caml.sourceforge.net/>

- F#

- [http://research.microsoft.com/  
projects/ilx/fsharp.aspx](http://research.microsoft.com/projects/ilx/fsharp.aspx)

