STT and the ML Programming Language

Kevin Everets
December 2, 2002

Overview

- A Brief History of ML
- ML and STT
- Ocaml and Some Examples
- Real World Usage
- Conclusion

A Brief History of ML

- ML: ``Meta-Language'' (not really)
- Developed in late 1970's at Edinburgh University for LCF Proof Assistant
- ~1984 Caml: Caml-Light & Ocaml (INRIA, Fr)
  - Categorical Abstract Machine Language
- ~1997 SML (Standard ML): SML/NJ & mosml

ML and STT

- Typing very similar, eg in ML:

let *add* x y = x + y ;;

val add : int -> int -> int = <fun>

- Similar to:

(lambda x,y: i. x + y)

(i -> (i -> i))

ML and STT (cont'd)

true ;;

- : bool = true

not **true** ;;

- : bool = false

let *compose f g x = f(g(x))* ;;

val compose : ('a -> 'b) -> ('c -> 'a) -> 'c -> 'b = <fun>

compose fact succ 8 ;;

- : int = 362880

ML and STT (cont'd)

- Main Types: Boolean, Integer, Float, Char, String, Unit (like **void**), Tuples (and Records), and Lists
- Possible to define more types; 2 major families:
  - *product* types for tuples or records
  - *sum* types for unions
- For example, creating a complex type:

type *complex* = { *re: float; im: float* } ;;

type complex = { re: float; im: float }

Ocaml

- Safe: compiler performs many sanity checks
- Static types, but compiler infers types!
- Fully Functional: Can pass functions as arguments
- Automatic Memory Management & Garbage Collection
- Efficient Compiler: can generate native or byte code.
- Object Oriented, Powerful Libraries, and so much more!

Some (more) Examples

- Recursive functions easy to express:

let rec *fact n* = **if** *n* = *0* **then** 1 **else** *fact*(*n*-1)*n;;

val fact: int -> int = <fun>

fact 8 ;;

- : int = 40320

- Equality's type:

(=) ;;

- : 'a -> 'a -> bool = <fun>

More Examples

- Identity Function:

```
# let id x = x ;;
val id : 'a -> 'a = <fun>
# id 3 ;;
- : int = 3
# id 3.2 ;;
- : float = 3.2
# id false ;;
- : bool = false
```

Real World Usage

- Many programs written in ML and Ocaml
- MMM: Web Browser
  - http://pauillac.inria.fr/mmm/eng.htm
- ICFP Contest: 2000 was to write a ray-tracer
  - http://www.cs.cornell.edu/icfp/
- MLDonkey: eDonkey2000 client (and more)
  - http://www.nongnu.org/mldonkey/

Conclusion

- ML is a good example of STT in practice
- It is actually very useful due to the compactness of the source code, the safety enforced by the compiler, and the efficiency of the resulting binary code
- STT is a Good Thing

References

- FOLDOC (Free On-Line Dictionary of Computing)
  - http://foldoc.doc.ic.ac.uk/foldoc/foldoc.cgi?ML
- O'Reilly Book: *Developing Applications with Objective Caml*
  - http://caml.inria.fr/oreilly-book/
- INRIA's Caml Pages:
  - http://caml.inria.fr/index-eng.html