Programming Languages, semantics, compilers

Christine Paulin
Christine.Paulin@universite-paris-saclay.fr

Département Informatique, Faculté des Sciences d'Orsay, Université Paris-Saclay

M1 Informatique - MPRI

2025–26

Tutorial

Ocaml

- Ocaml : used to define/describe concepts, data-structures and algorithms
- Undergraduate courses at Faculté des Sciences d'Orsay
  - Introduction à la prog. fonctionnelle (IPF)
    `https://usr.lmf.cnrs.fr/~kn/ipf_fr.html`
  - Programmation fonctionnelle avancée (PFA)
    `https://usr.lmf.cnrs.fr/~kn/pfa_fr.html`
- References :
  - Main site (installation, manual, exercises) `https://ocaml.org/`
  - See also IPF site (instructions for vscode use at PUIO)
  - Book "Apprendre à programmer avec Ocaml"
    S. Conchon et J.-C. Filliâtre.
    English version "Learn Programming with OCaml" libre
    `https://usr.lmf.cnrs.fr/lpo/lpo.pdf`

A little bit of History

- 1972 : ML (Meta-Language for LCF, arbres, typage, exception, abstraction...) by Robin Milner (1934-2010)
- at INRIA, Gérard Huet's team developed several versions of ML
  - 1884 : ML implementation based on (Le)LISP
  - 1987 : CAML compiler by Ascánder Suárez, Pierre Weiss and Michel Mauny
  - 1990 : Caml-light by Xavier Leroy and Damien Doligez (byte-code interpreter, garbage collector)
  - 1995 : Caml Special Light by Xavier Leroy (native-code compilation, modules)
  - 1996 : Objective Caml (Ocaml) adding objects (Didier Rémy and Jérôme Vouillon)
  - + other features

Overview of the main ideas

- Write and evaluate *expressions*
- Any expression comes with a *type*
- A variable is just a name for a *value*
- Explicit distinction between variables and *references*
- Functions as first-class citizens, loops as (tail-)recursive functions
- Data-structures :
  - tuples
  - records
  - arrays
  - lists
  - trees
- Association tables (as lists or hashtables)