À la découverte du langage OCaml

Xavier Van de Woestyne

LilleFP6

Septembre 2017

WHOAMI

- Xavier Van de Woestyne
- **xvw** sur Github et **vdwxv** sur Twitter
- Développeur chez **Fewlines** (Elixir, Elm)
- Erlang/**Elixir**, **OCaml**, F#, Haskell, **Elm**, Ruby, **Nim**, Scheme etc.
- J'aurai un blog (https://xvw.github.io)

LilleFP : pour des rencontres techniques

- **Généralisation** des sujets des événements
- Mutation de **l'ambiance** générale
- Ouvert à beaucoup de sujets **techniques** (Pourquoi pas vous ?)
- Rejoignez-nous sur **Slack** : https://slackin-lillefp.herokuapp.com/

Merci à nos partenaires (de tous temps)

Fewlines, Dernier Cri, Mozilla foundation, Synbioz, Epitech Lille et TakeOff talks.

Pourquoi cette présentation ?

Pour réfuter cette affirmation :

> *OCaml est un langage dédié à la recherche.*

Et pour enrichir cette affirmation :

> *OCaml est un bon langage pour faire de l'analyse et des compilateurs.*

Car, OCaml est un langage **très très** polyvalent et c'est donc intéressant d'en survoler rapidement les fonctionnalités.

Malheureusement

Comme je ne suis pas illimité en temps :

- Je survolerai les fonctionnalités du langage
- L'objectif de la présentation est de stimuler un **intérêt**
- Je ferais avec joie un Workshop OCaml/ReasonML

Qu'est ce que, rapidement, OCaml

Un langage :

fonctionnel, impératif, typé fortement et statiquement, compilé/inteprété, portable et performant, issu de la recherche.

Utilisé dans l'industrie :

Airbus, Tezos, Facebook, Mozilla, Docker, JaneStreet, Citrix etc.

Qui a inspiré :

F#, F*, Ur, Rust, Reason, Hack, Swift... RML, etc.

Success-Stories

MirageOS, Coq, MIDonkey, Tezos, aMSN, Unison, HHVM, Infer, Flow, F* et beaucoup d'autres !


Companies utilisant OCaml

Pourquoi apprendre OCaml

- Etendre votre boite à outils
- Ouvrir de nouvelles perspectives (pour d'autres langages)
- Devenir polyglote

OCaml m'a permis de devenir un meilleur programmeur Ruby.

Sommaire

1958 Lisp

- Développé par J. McCarthy
- Impératif et fonctionnel
- Typé dynamiquement
- Premier langage moderne
- Beaucoup d'enfants (Common Lisp, Scheme, Arc, Racket, Clojure, Emacs Lisp)

De ML à Reason

1970 ML (Meta Language)

- Développé par R. Milner et son équipe de l'université de Edimbourg
- Impératif et fonctionnel
- Typé statiquement (pour le système de preuve LCF)
- Inférence de types, correspondance de motifs et modules puissants

De ML à Reason

1983 SML (Standard ML)

- Une standardisation de ML
- Plusieurs implémentations (SML of New Jersey, Moscow ML, MLTon, Poly/ML)

De ML à Reason

1985 Caml (Categorical Abstract Machine Language)

 A History of Caml

- Implémentation Française de ML
- Développé pour compiler vers une machine Lisp (LLM3)

De ML à Reason

1990 Caml Light

- Nouvelle implémentation par Xavier Leroy (entre autre)
- Compilant vers un byte-code interprété en C
- Portable
- Avec un très bon garbage collector

De ML à Reason

1995 Caml Special Light

- Ajout d'une compilation vers du code natif
- Performances compétitives (proche de C++)
- Système de module très évolué inspiré de celui standardisé dans SML
- Utilisé pour enseigner la programmation en Prépa, jusqu'il y a peu...

De ML à Reason

1996 OCaml (Objective-Caml)

- Programmation orientée objets (avec inférence) *type-safe*
- 2000 : Ajout de fonctionnalités expérimentales, variants polymorphiques, arguments optionnels, labellisés, méthodes polymorphiques etc.
- Performances encore acrues

De ML à Reason

2002 F# (comme C# mais avec un F pour fonctionnel)

- Avec un noyau dérivé de OCaml
- Syntaxe allégée
- Compatible .NET
- Pas de modules au sens ML
- Expressions de calcul

De ML à Reason

2015/2016/2017 ReasonML

- Un nouvelle syntaxe pour OCaml
- Ajout de pleins de features modernes “à la Elm”
- Focus sur les outils
- Pour plaire à la communauté JavaScript

OCaml est donc un langage assez ancien, qui a eu l'occasion d'être éprouvé !

Outils

- **ocamlc** : compilateur vers le byte-code
- **ocamlopt** : compilateur vers du code natif
- **opam** : un gestionnaire de paquet (standard)
- **merlin** : IDE-features pour tous les éditeurs modernes (Emacs, VSCode, Atom, Sublim Text et … euh veem).
- **js_of_ocaml** : compilateur JavaScript
- **BuckleScript** : transpileur JavaScript
- **gen_js_api** : un créateur de binding OCaml <-> JavaScript

Et beaucoup d'autres

Camlp4, Lwt, et pléthore de build-system (**ARGH**).