

Un aperçu du langage CAML

CAML est un langage de programmation *généraliste*, conçu pour garantir la *sûreté* et la *fiabilité* des programmes.



CAML appartient à la famille des langages *fonctionnels fortement typés*.

Un langage fonctionnel

La programmation fonctionnelle est un paradigme de programmation qui considère le *calcul en tant qu'évaluation de fonctions mathématiques*. Elle met en avant la définition et l'application de fonctions. Les fonctions sont des valeurs ordinaires : une fonction peut être passée en argument à une fonction ou renvoyée en tant que résultat par une fonction.

Remarque 1

On peut essayer de faire de la programmation fonctionnelle en PYTHON¹, mais le langage s'y prête moins bien.

Un langage fortement typé



En CAML chaque expression appartient à un type donné, sans ambiguïté possible.

CAML dispose d'un système de types puissant, doté de polymorphisme paramétrique, d'inférence automatique de types, de types de données algébriques définissables par l'utilisateur et d'un filtrage élégant et concis.

Sûreté

Le langage CAML est sûr. Le compilateur effectue une vérification complète des programmes avant de permettre leur exécution. Ceci interdit de nombreuses erreurs de programmation, par exemple la confusion entre un entier et un pointeur ou l'accès à un champ inexistant d'un enregistrement.

OCAML

Le langage officiel² retenu pour l'enseignement de l'option informatique est OCAML (c'était CAML LIGHT jusqu'en 2018).

OCAML étend le noyau du langage CAML avec une couche *orientée objets* complète, ainsi qu'un système de modules puissant. Sauf éventuellement si vous utilisez CAML pour votre TIPE, nous ne nous servons pas de ces fonctionnalités avancées.

Le système OCAML est un logiciel libre : le compilateur est distribué sous les termes de la Q Public License, et sa librairie sous ceux de la LGPL.

1. Voir par exemple <https://fgallaire.flext.net/tag/programmation-fonctionnelle/>, à réserver à ceux qui sont vraiment très à l'aise.

2. http://www.education.gouv.fr/pid285/bulletin_officiel.html?cid_bo=123904

Traits impératifs

CAML n'est pas un langage fonctionnel pur³ et permet la programmation impérative. En particulier, variables, tableaux, et composantes des enregistrements peuvent être déclarés modifiables et on peut définir des boucles.

Gestion de la mémoire automatisée

Comme PYTHON, CAML dispose d'une gestion automatique de la mémoire grâce à un ramasse-miette (en anglais *garbage collector*) efficace. Cela rend les programmes beaucoup plus sûrs : toute corruption de la mémoire est impossible.

Un langage de programmation largement utilisé

Le langage CAML a initialement été utilisé pour développer des applications réalisant des calculs symboliques : prouveurs de théorèmes, compilateurs et interprètes, analyseurs de programmes, etc. Il est maintenant utilisé pour réaliser toutes sortes de logiciels dans de nombreux domaines d'application. Plusieurs grosses entreprises développent des applications industrielles significatives⁴ en CAML, par exemple Dassault Systèmes, Jane Street, Microsoft, IBM et le CEA.

Un exemple : ASTRÉE est un analyseur statique basé sur l'interprétation abstraite et qui vise à établir l'absence d'erreurs d'exécution dans des logiciels critiques. À ce jour, ASTRÉE a prouvé l'absence d'erreurs d'exécution dans le logiciel de contrôle primaire de la famille Airbus A340. Cela serait impossible par de simples tests, car le test ne considère qu'un sous-ensemble limité des cas, tandis que l'interprétation abstraite considère un sur-ensemble de tous les comportements possibles du système.

Une petite histoire de CAML

CAML est l'acronyme de *Categorical Abstract Machine Language* :

- La CAM était une machine abstraite capable de définir et d'exécuter les fonctions (basée sur une correspondance entre lambda-calcul et combinateurs catégoriques) (1984).
- Le ML était le *meta-langage* d'un assistant de preuve automatique conçu par Robin Milner en 1978.

La première motivation au développement de CAML était de pouvoir l'utiliser pour développer l'assistant de preuve Coq⁵.



- (1987) Première implémentation par Ascander SUAREZ, Guy COUSINEAU, Pierre WEIS et Michel MAUNY ;
- (1990) CAML LIGHT, nouvelle implémentation par Xavier LEROY, qui aida beaucoup à répandre l'emploi de CAML dans l'enseignement et la recherche ;
- (1995) CAML SPECIAL LIGHT, ajout d'un compilateur produisant du code natif et d'un système de modules de haut niveau ,
- (1996) OCAML, ajout d'un système de type pour les objets et les classes permettant la programmation par objets ;
- (2005) F#, langage de programmation fonctionnel, impératif et orienté objet dérivé du langage OCAML, pour la plate-forme .NET, développé par MICROSOFT RESEARCH.

Pour en savoir plus : <https://caml.inria.fr/about/history.fr.html>.

3. Comme langage fonctionnel pur on peut citer HASKELL (<https://www.haskell.org>).

4. Voir <https://ocaml.org/learn/success.fr.html>.

5. <https://coq.inria.fr>