Programmation Fonctionnelle

Delia Kesner
kesner@irif.fr
https://www.irif.fr/~kesner/

Intervenants

CM: Mercredi 14h-16h: Delia Kesner
kesner@irif.fr

TP: Lundi 10h45-12h45: Vincent Padovani
padovani@irif.fr

TP: Mardi 10h45-12h45: Giovanni Bernardi
gio@irif.fr

TP: Mercredi 16h15-18h15: Jéremy Ledent
jeremy.ledent@irif.fr

TP: Jeudi 10h45-12h45: Clement Allain
clement.allain@orange.fr

TP: :Vendredi 10h45-12h45 Alexandre Moine
alexandre.moine@inria.fr

Organisation

- Les TP commencent le mercredi 27 septembre
- Examen final : en janvier
- Projet de programmation obligatoire
  - en binôme
  - date de remise (unique): début décembre
  - absence $= 0$
  - plagiats $= 0 +$ procès-verbal
  - Plus d'information en TP

Contrôle de connaissances

- Première session :

$$\frac{1}{3} * \text{projet} + \frac{2}{3} * \text{exam1}$$

- Deuxième session :

$$\max(\frac{1}{3} * \text{projet} + \frac{2}{3} * \text{exam2}, \text{exam2})$$

Organisation

- Support : transparents du cours et plusieurs ressources en ligne
- Indispensable d'assister au cours et au TP
- Des informations importantes sont envoyées regulièrement sur la liste du cours `https://listes.u-paris.fr/wws/subscribe/l3.pf5.info`.

Inscrivez-vous vite!

Deux styles de pogrammation

Programmation impérative:

- Programme: séquence structurée d'instructions modifiant létat de la machine
- Évaluation: exécution des instructions les unes après les autres jusqu'au point de sortie

Programmation fonctionnelle:

- Programme: expression définissant une fonction
- Évaluation: réécriture d'une expression jusqu'à l'obtention d'une valeur

Exemple de programme impératif

Le plus grand commun diviseur:

```
static int pgcd(int m, int n){
int aux;
while (m != 0) {
    aux = m;
    m = n % m;
    n = aux ;  }
return n ;
}
```

Résultat: 2

m	n	aux
4	6	
4	6	0
4	6	4
2	6	4
2	4	4
2	4	2
0	4	2
0	2	2

Exemple de programme fonctionnel

```
let rec pgcd(m, n) =
if m=0 then n
   else pgcd(n mod m, m)
```

$$
\begin{array}{ll}
\texttt{pgcd}(4,6) & \rightarrow \\
\texttt{if } 4=0 \texttt{ then } 6 \texttt{ else pgcd}(6 \texttt{ mod } 4,4) & \rightarrow \\
\texttt{if false then } 6 \texttt{ else pgcd}(6 \texttt{ mod } 4,4) & \rightarrow \\
\texttt{pgcd}(6 \texttt{ mod } 4,4) & \rightarrow \\
\texttt{pgcd}(2,4) & \rightarrow \\
\texttt{if } 2=0 \texttt{ then } 4 \texttt{ else pgcd}(4 \texttt{ mod } 2,2) & \rightarrow \\
\texttt{if false then } 4 \texttt{ else pgcd}(4 \texttt{ mod } 2,2) & \rightarrow \\
\texttt{pgcd}(4 \texttt{ mod } 2,2) & \rightarrow \\
\texttt{pgcd}(0,2) & \rightarrow \\
\texttt{if } 0=0 \texttt{ then } 2 \texttt{ else pgcd}(2 \texttt{ mod } 0,0) & \rightarrow \\
\texttt{if true then2 else pgcd}(2 \texttt{ mod } 0,0) & \rightarrow 2
\end{array}
$$

Résultat: 2

La Programmation Fonctionnelle

- Les fonctions sont des données de première classe: elles peuvent être retournées en résultat et passées en argument à d'autres fonctions
- Permet d'écrire des programmes qui sont plus proche au problème donné
- Favorise la vérification (même automatique) des comportements des programmes
- ... et cela sans perte importante d'efficacité

Différents langages fonctionnels

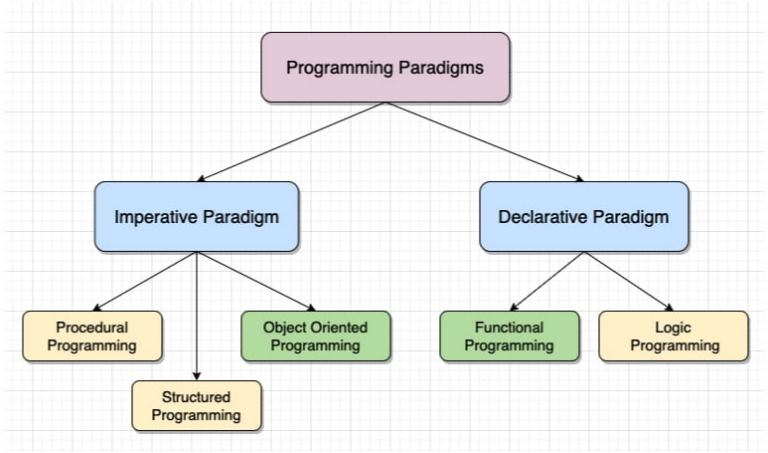
On distingue :

- les langages fonctionnels *purs* (Haskell...)
- les langages qui combinent fonctionnel et impératif (OCaml)

Caracteristiques majeures

- Fonctions récursives travaillant sur des structures de données inductives (listes, arbres, ...) ;
- Contrôler voire à supprimer toutes formes d'état
- Schémas de calcul (itération, accumulation, filtrage, composition, réduction, ...) implémentées par des fonctions d'ordre supérieur

Les paradigmes de programmation (concis)

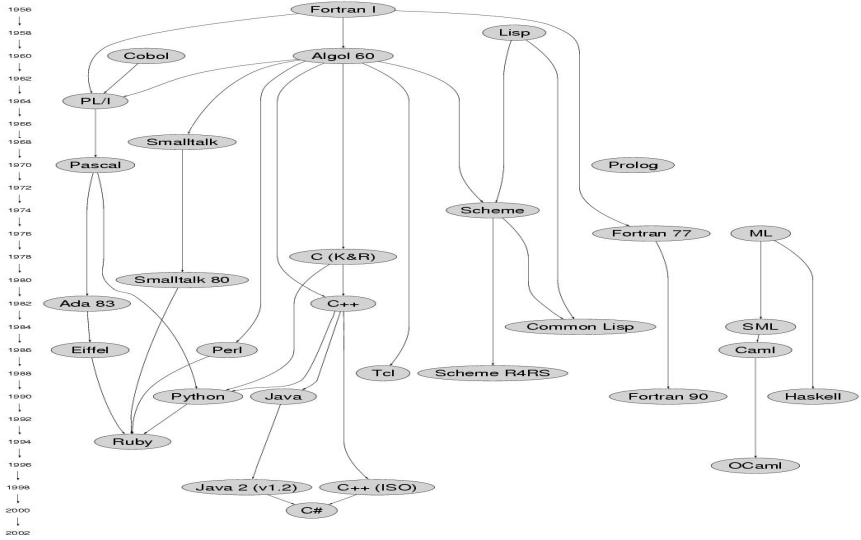


Fonctionnels	Impératifs	Orientés Objet	Déclaratifs/Logiques	Concurrents
193x-λ-calcul				
	1956 Fortran			
1958 Lisp				
	1959 Cobol			
	1963 Basic			
	1970 C			
			1972 Prolog	
197x Scheme				
	197x Pascal			
	198x C++			
	1983 Ada			
1985 Miranda				
1987 Caml				
				1987 Erlang
...1990 Haskell				
		199x Java		
	1996 OCaml			
		2000 C#		
	200x F#			

Histoire (très synthétique) de OCaml

- Années 70: Robin Milner propose ML comme méta-langage de l'assistant à la preuve LCF d'Édimbourg
- 1980: Projet Formel à l'INRIA (Gérard Huet), types de données algébriques et le filtrage (Guy Cousineau), Categorical Abstract Machine (Pierre-Louis Curien)
- 1984-1990: Définition de SML (Milner)
- 1987: *Caml* (implémenté en Lisp) Guy Cousineau, Ascander Suarez, (avec Pierre Weis et Michel Mauny)
- 1990-1991: *Caml Light* par Xavier Leroy (et Damien Doligez pour la gestion de la mémoire)
- 1995: *Caml Special Light* puis 1996 *OCaml* (Xavier Leroy, Jérôme Vouillon, Didier Rémy, Michel Mauny)
- 2011: Le nom Objective Caml (OCaml) est adopté.

Dans le temps



Spécificités d'OCaml

Haut Niveau:

- fonctions comme objets de première classe (fonctionnel)
- fonctions mécanisme d'abstraction puissant
- gestion de mémoire automatisée (*Garbage Collector*)

Multi-paradigme:

- styles fonctionnel, impératif, orientée objet
- graphisme, applications réseaux, ...

Système de typage:

- permet reutilisation du code grâce au polymorphisme
- types synthétisés automatiquement par le compilateur
- capture beaucoup d'erreurs

Performance:

- Génération de code efficace

Quelques sites/livres en ligne

- Ces notes sont basées sur les cours précédents de R. Treinen, M. Pagani, Y. Régis-Gianas, P. Letouzey
- `http://ocaml.org/`
- `https://www.fun-mooc.fr/fr/cours/introduction-to-functional-programming-in-ocaml/`
- Yaron Minsky and Anil Madhavapeddy *Real World OCaml - Functional Programming for the masses*
  `https://realworldocaml.org`
- Emmanuel Chailloux, Pascal Manoury and Bruno Pagano: *Developing Applications With Objective Caml*
  `https://caml.inria.fr/pub/docs/oreilly-book/`
- Xavier Leroy et al : *The Objective Caml system : documentation and user's manual*
  `https://caml.inria.fr/pub/docs/oreilly-book/ocaml-ora-book.pdf`

Autres ouvrages

Guy Cousineau et Michel Mauny
Approche fonctionnelle de la programmation
Dunod, 1995

Pierre Weis et Xavier Leroy
Le langage Caml
Dunod, 1999

Catherine Dubois et Valérie Ménissier-Morain
Apprentissage de la programmation avec OCaml
Hermès, 2004

Louis Gacogne
Programmation par l'exemple en Caml
Ellipse, 2004

Autres ouvrages (suite)

Philippe Nardel
Programmation fonctionnelle, générique et objet : Une introduction avec le langage OCaml
Vuibert, 2005

Joshua B. Smith
Practical OCaml
Apress Berkeley, 2008

John Whitington
OCaml from the Very Beginning
Coherent Press, 2013 (disponible à la bibliothèque centrale)

Sylvain Conchon et Jean-Christophe Filliâtre
Apprendre à programmer avec Ocaml
Eyrolles, 2014