Tutoriel 2 - Coq, un OCaml pur

Déclaration	OCaml	Coq

Type	`type t =`
	`  | A of bool`
	`  | B of nat`
		`Inductive t :=`
		`  | A : bool → t`
		`  | B : nat → t`

Fonction Fonction récursive	`let f = ...` `let rec f = ...`	`Definition f := ...` `Fixpoint f := ...`
Pattern-matching	`;;` `->`	`end` `=>`

Note : En OCaml, on écrit = et, en Gallina, on écrit `:=`.

Coq est ...

- ... très similaire à OCaml ...
  - Fonction récursive
  - Pattern-matching (filtrage)
  - Polymorphisme
  - Les fonctions sont des valeurs de première classe.
  - Application partielle
  - Fonction anonyme
- ... mais plus rigide que OCaml ...
  - **Une fonction doit TOUJOURS terminer !**
    Cas détecté automatiquement par Coq : **récurrence structurelle**
  - **Un pattern-matching est FORCEMENT exhaustif et non-redondant.**
  - Pas d'exception
- ... mais aussi plus souple et plus expressif !
  - Définition inductive (type inductif, prédicat inductif)
  - Type dépendant
  - Paramètre implicite
  - Déclaration de notation