Saroupille

Zeste de savoir

Faut-il *ghoster* les noms de variable ?

21 janvier 2019

Table des matières

1. Introduction

En programmation fonctionnelle, il m'arrive souvent d'écrire ce type de code :

```
let x = f(...) in
let x = g(...) in
let x = h(...) in
...
```

Les deux premières variables deviennent alors inutilisables car elles sont cachées par la dernière déclaration. Mais en général, ce n'est pas un soucis car on en a pas besoin par la suite, ce sont juste des valeurs temporaires. Mais est-ce une bonne pratique ?

Premièrement, pour des raisons d'efficacité, on ne peut pas vraiment se passer des variables temporaires, et même si on le pouvait, en général, cela donne un code à rallonge très difficile à lire.

Deuxièmement, imaginons que j'ai donné des noms différents :

```
let x = f(...) in
let x' = g(...) in
let x'' = h(...) in
...
```

Alors le corps de la fonction peut parler de `x`, `x'` ou `x''`. Or, au début, ce qui m'intéressait, c'était seulement `x''`. Cela peut donc être source d'erreur dans mon code qui parfois s'avère compliqué à repérer.

Je m'autorise donc de changer les noms seulement si les types de `x`, `x'` et `x''` sont différents. Ainsi, si je me suis trompé, je me prendrai un avertissement de la part du *type-checker*. Si les types sont les mêmes, alors il me semble que c'est une bonne pratique de *ghoster* les noms de variables pour éviter ce type de problème.

Un autre cas où il m'arrive de masquer le nom d'une variable est lorsqu'une fonction a besoin d'une fonction auxliaire :

```
let pp_list pp fmt list =
  let rec pp_list pp fmt = function
  | [] -> Format.fprintf fmt "]"
  | [x] -> Format.fprintf fmt "%a]" pp x
  | x::t ->  Format.fprintf fmt "%a;%a" pp x (pp_list pp) t
  in
  Format.fprintf fmt "[%a" (pp_list pp)  list
```