Abstractions; Haskell vs Clojure

How are both Haskell and Clojure good to capture some abstractions

Yann Esposito

[2020-01-18 Sat]

This is a short essay to try to find if Haskell is as limited in its abstractions power as I experienced many times when learning it. Now I've got a lot more experience in Haskell and in Clojure. I would like to see if I'll be able to manage to put some abstractions in Haskell and in Clojure.

Abstractions

The notion of abstraction is generally not pretty well defined.

For example, not using pointer can be considered an abstraction. Garbage collection is an abstraction. Procedures are an abstraction. ORM are an abstraction. Monads are an abstraction. Objects and Classes are an abstraction.

I would be surprised is not anyone has made a kind of hierarchy to organize different abstractions as they seems to me of different nature.

In this post I will only focus on those few abstractions:

- Object Oriented Programming (Objects and Classes, inheritence, etc...)
- Frameworks for managing the lifecycle and dependencies of software components with have runtime state (in Clojure: Component or Trapperkeeper)
- ORM/Store abstractions

Object Oriented

So in Clojure, this is quite straightforward if you consider you are authorized to use the underlying language (java).

Haskell

Notion of Object, something that contain an inner state + some methods. Notion of Class, a boilerplate for creating new Objects.

```haskell
newtype MethodName = MethodName Text
data Object s m = Object { state :: s
                         , methods :: forall a. MethodName -> s -> a }
```

hmmm.... At the very begining of trying to write it down, it already suck...