Programming experiences and choices

Yann Esposito

TODO Introduction

Why each programmer tend to prefer some programming language to solve its problems. How are we creating our preferrences? Why one use vim and the other one can use a specialized IDE?

Preferences depends on experiences

Hard to understand

Goals

Back Story

At first

- Logo when I was 10yo at school
- Basic when I was 11yo, with a book trying to draw lines and make games
- Basic with Amtrad CPC 6128, trying to write games from magasines
- Compiled Basic with Atari STe, write a game you are the hero in it.
- Take some courses of beginner Pascal at school, sort algorithms

Math Background then Computer Science

- Pascal for algorithmic
- C, for basics, system and network
- C++, Eiffel for Object Oriented Programming
- A little bit of CaML (write a mathematical expression simplifier making big usage of pattern matching)

Ph. D. In Machine Learning

- Give courses of Logo, C, etc...
- Write lot of complex HMM related algorithms in C++ with quite complex ML algorithms
- Discover Java and its promises, play a bit with it

- Have friend that use CaML for its Ph.D. and its hash-maps are said to be faster than C. I remember this.

No Love for Machine Learning in 2005 :/

- Write a Java program with an User Interface in my Post Ph. D. 1/2 of the work in the UI, the other half in the algorithm.

Still no love for ML in 2006 Find a job just to eat

- Go find a job to eat, have a **lot** of time to learn new things
- Discover HN, /r/programming, etc...
- Web Applications are all the rage
- DCVS is still a thing, people argue between git, mercurial, bazaar, etc...
- Write a tremendous number of zsh scripts to handle a huge number of files, use Perl, etc...

First try at startup

- Decide with a friend to make a product, choice of technology with them is very hard.
- git for example was a question
- using FB connect instead of classical, name / password bullshit is refused because they still live technically in 2000
- Design decisions are hard to make
- Programming language, I heard good things about OCaML as the fastest high level language. Can't even talk about it.
- A guy need technicians to make its product and is willing to pay.
- OK good first thing to try ourselves.
- The other guy know Ruby, so let's go with Ruby (no rails)
- Write our own framework, many technical frictions, but in the end a PoC is made full ruby, deployed on heroku. The product is an end-to-end personal electrical power consumption system.
- Guy explain, I want ”real time”!!!! ???? WTFBBQ!! Real time is way harder than just drawing dashboards!!!!
- Have performance problems! Start looking into other frameworks, stumble upon snap, an Haskell framework stating that it is very fast.
- Start looking further into Haskell from there.

Haskell learning

- Learn Haskell from web programming perspective, the goal is not to learn Haskell but to use Haskell to write as fast as possible a web application with batteries included. After trying a bit, I choose to use Yesod.
- Lot of time lost due to Yesod difficulties to handle correct package version coherence!!!!

- cabal freeze, etc... to the rescue, not perfect but ok, able to deploy on heroku
- Learn Haskell in the process.