Philosophy of Sciences
Philosophy and esoteric programming languages

Bruno Rocha Pereira - 529512

June 4, 2016

Word count: 1947

1 Introduction

Esoteric languages are programming languages that were created for no particular reason but the recreational value to the creator and the entertaining factor for the user. They mostly exist as a joke but can also be used to test the boundaries of a certain programming paradigm or to link programming to art in a uncanny way. This paper will study the beneficial contributions such languages can bring to the life of any scientist and the links existing between those and philosophy. A succint history of the first esoteric languages is also given as a base for this small study and some more recent languages will be studied more in depth.

2 Programming is philosophy

While completely unrelated at first sight, Computer science is a actual exact science and the philosophy of science is a branch of the general philosophy[1]. This relates both of them in an uncanny but still Programming is indeed, as thinkers from the past – including Aristotle, Hobbes and Turing – hoped would exist, a way to automate reasoning. Such a process requires logical rigour, and think about the consequences of new ideas, pushing further and further the bounds of knowledge and the nature of reality.[2] Thinking about the nature of a program and its evolution is a philosophical quest.

Some developers even believe that "Plato laid the foundation for what is now Object-Oriented Programming, while Aristotle's dynamic model is at the core of FP."[3]. They take this assumption from the Raphael's "School of Athens" famous painting, analyzing their hand gestures to deduce that Plato's and Aristotle's

[1] http://cas.umkc.edu/philosophy/vade-mecum/branches.htm

[2] https://www.ox.ac.uk/admissions/undergraduate/courses-listing/computer-science-and-philosophy?wssl=1

[3] https://www.infoq.com/presentations/Philosophy-Programming

point of view and how they influenced modern programming. A comparison is made between Plato's Forms and the Physical world, and the Classes and Objects from Object-oriented programming. Both the Forms and Classes are what is "Real", what would be an ideal state, while both the Physical world and the objects are the most concrete part. On the other hand, Functional Programming is based on the transformations, transitions (becoming) which can be compared to Aristotle's concept of Hylomorphism.

3 Brief history of the esoteric languages

The very first esoteric language to be ever designed was `INTERCAL` [4], the obvious acronym for "Compiler Language With No Pronounceable Acronym"[5]. The first goal of it was to make a satire of the different programming languages. Computer scientists almost ignored the field of esoteric languages until 1990, which saw the revival of `INTERCAL` in the programming language C. From then, more and more languages started being created. This was the case for the language FALSE which goal was to "confusing everyone with an obfuscated syntax, and designing an as powerful language as possible with a tiny implementation"[6]. The esoteric language Brainfuck was then also invented, with a clear intention that made its name. A lot of languages were developed with different approach later.

4 Diversity

Almost any idea could be transformed into an esoteric language. Some of them are inspired by art, like for example `Piet`, inspired by the work of the Dutch painter Piet Mondrian or `Velato`[7] that uses MIDI files as input which allows the user to program with music. They sometimes also take inspiration in literature, like `The Shakespeare Programming Language` [8] or from internet memes like `LOLCODE` [9]. Besides the diversity of the sources of inspiration, the different languages also have different purpose. Some of them aim exploring alternative concepts while others try to have a shot at being the weirdest or the funniest language.

Esoteric languages are still being invented and implemented day after day and the most famous website lists a fair amount of ideas that are only waiting for an implementation of the actual language[10]. These ideas concern very diverse categories, such as "Silly ideas", derivation from existing languages - esoteric or

[4] `http://catb.org/esr/intercal/`
[5] Note from the writer : I did not make this up.
[6] `http://strlen.com/false-language`
[7] `http://danieltemkin.com/Velato`
[8] `http://shakespearelang.sourceforge.net/report/shakespeare/`
[9] `http://lolcode.org/`
[10] `http://esolangs.org/wiki/List_of_ideas`

not- , Music, Mathematics or Game-based languages. All of those ideas could lead to new esoteric, Turing-complete or not, languages.

5 Usability

The usability of that kind of languages is very restricted. Most of them are only usable to display a useless feature, implement a simple `Hello World` program or push programming to another pointless level. None of those languages have actually ever been used to create real software. The main reason is that most - if not all- of them are Turing tarpits. That means they offer the bare minimum of functionalities or use a very terse notation[11], limiting their overall usability and the will of any programmer to use that language in order to develop softwares with them.

6 Real Goal

Despite the fact that most of the esoteric languages have no usability, they allow anybody that is interested in it to trick his own mind and have a new approach to programming languages. They aim at creating new concepts with the minimal set of instructions possible. These concepts can nonetheless bring a whole new vision on programming. The question that can be and usually is asked is why are those created. Not all of them have an actual purpose but having fun while creating or discovering it. Yet, they can bring a real challenge to the developer of it and any eventual discoverer.

Besides the fun factor they bring, such languages can bring a whole new thought process. They can defy all logic rules that were previously assumed by any computer scientist or philosopher and replace it by the creator's own original way of thinking with completely antagonistic rules. This gives an outer vision of computer sciences, completely different than the one computer scientists are used to see. That external view broadens the field of knowledge and the vision of the concerned. It indeed forces the user to open his mind and get off the beaten track of classic programming languages.

7 Esoteric languages and philosophy

As said in the sections above, anything can be used as a basis for an esoteric language. Philosophy and philosophers were also used as inspiration for some esoteric languages. This is the case for example of `Stasis`[12], `Philosophy Script`[13] and `Wittgen`[14].

[11] `http://c2.com/cgi/wiki?TuringTarpit`
[12] `http://esolangs.org/wiki/Stasis`
[13] `http://esolangs.org/wiki/Philosophy_Script`
[14] `http://esolangs.org/wiki/Wittgen`

7.1 Stasis

Stasis is described as a "pre-Socratic language first produced in the 5th Century BCE" and its logic is based on a quote from the famous pre-socratic philosopher Parmenides:

> We can speak and think only of what exists. And what exists is uncreated and imperishable for it is whole and unchanging and complete. It was not or nor shall be different since it is now, all at once, one and continuous...

From it are extracted a few basic rules that create the language:

- The universe is in constant balance
- There is only one number, its value is zero
- Other numbers are illusory, refractions of zero

Stasis only replaces the basic structures logic from C#, which means it could actually be used to make real softwares using all the rest of the features from that language. Developing such a program with it would still be a challenge since the base logic is changed, which affects all the rest of the structures.

7.2 Philosophy Script

Philosophy Script does not take inspiration in a particular philosopher but uses instead philosophical questions and philosophers name as a set of instructions. Here are a few example commands:

- `Why x=1?` assigns the value 1 to the variable x
- `presocratic.Anaximines[This is text]` displays text to the screen
- `If (x=1) moral [ commands ] debate [ commands ]` represents a standard if...else... structure where the block of instructions executed when the condition is true is called the moral and the opposite
- `Can we know anything?` deletes a file
- `Why is there suffering?` deletes all files in the computer

Many

7.3 Wittgen

The main goal of Wittgen's development is to provide answers to some of the problems Wittgenstein presented in his *Philosophical Investigations*. Wittgenstein presents the philosophical concept of Language-game refering to the way a language is used. This of course originally applies to the communication between humans, but can however be extended to the communication between a human and a computer. Programming can be seen as a language game. Wittgenstein describes a game as working because it can be learned. He also qualifies a game as having that seems to be applying rules while not specifically having rules. The notion of explicit rules is confused. This applies perfectly to esoteric languages since they clearly make a game of language where concepts do not need to be explicitly defined - and rarely are - to be meaningful and define a language. In the quote:

> The language is meant to serve for communication between a builder A and an assistant B. A is building with building-stones: there are blocks, pillars, slabs and beams. B has to pass the stones, in the order in which A needs them. For this purpose they use a language consisting of the words "block", "pillar" "slab", "beam". A calls them out; — B brings the stone which he has learnt to bring at such-and-such a call. Conceive this as a complete primitive language. [15]

The assistant B could be easily replaced by the computer and the builder A would be split into 2 roles. The first role would be the language designer who would define the language and "teach" it to both other roles, while the second role would be the user that actually has to use the defined language. This again proves that there can be an intricate link between philosophy and esoteric languages.

8 Personal opinion

As useless as esoteric languages appear, they all bring a unique approach to programming. It's interesting to see that some of them have an ingenious approach. As said earlier, the extent of any esoteric language is only limited by the creativity of the developer. This leads to the creation of all the kind of incredibly ingenious languages like `Velato` or `Piet`. They are actually not as easy to implement as one could think and can be hard to master for the users. But most of them offer a test and entertainment that regular programming languages don't usually offer. Of all those languages, `INTERCAL` is probably the most auto-obfuscating one. It parodies several languages at the same time which results in the impossibility to use any basic functionality in a clear and easy way.

Esoteric languages also offer a fresh vision on programming itself. They indeed show that a regular language isn't as easy as some may think. Any programming language must respect the standards of double-coding in order to be usable, which means they have to be understandable at the same time by the user and the computer. They indeed show that it's close to impossible to program in a language that is barely readable by the user. It also leads to confuse any experienced programmer that is used to known and overall structures and gets easily used to any language-related feature. Esoteric languages do not have

[15]Ludwig Wittgenstein - Philosophical Investigations §2, 1953

any notion of how the code should be "well structured", which can also be confusing for the users when they try to implement anything correctly in that language.

9 Conclusion

Esoteric languages bring a whole new approach to the traditional ways of programming, exploring new ways and creating new challenges for the content developers or for the users. They give a different approach to programming than standard programming languages while not bringing any usability. Interestingly enough, esoteric languages can be compared to certain parts of philosophy, whether as a whole or taking few examples of those languages.