The role of Aesthetics in the Understandings of Source code

Pierre Depaz
under the direction of Alexandre Gefen (Paris-3)
and Nick Montfort (MIT)

ED120 - THALIM

last updated - 04.10.2021

1 Introduction

This thesis stems from a desire to better understand the aesthetics specific to source code, and through this, to flesh out our understanding of what source code really is. It may look like text, but it's not text. It may look like something hidden and unnecessary, but it's not. It is something that programmers interact with daily, but remains hidden from view, with litterature often referring to code as one, general, generic concept, and do not take into account the specificities of writing and reading code. I wanted to examine the intricacies of code, something that happens through concrete manifestations, focusing on an empirical approach not just theories or abstract conceptualizations. In a nutshell, people talk about it a lot, and I want to ask: what does it *actually* look like? What I also wanted to learn about is the relationship between beauty and function. Since code does, and since source code's purpose is to disappear, how could something which is written off as temporary, hidden, still hold meaningful aesthetic manifestations? Could it be that there is beauty just for its own sake, or beauty for the sake of something else. Related to this, there is the polymorphous nature of code, and of the communities around code. There isn't just one code, but there are a bunch of different ways to write code, from poems to apps to hacks to theory. So I was curious about whether or not there is something *at the root of code* that is common to all these communities. It might end up being something like double-coding. Or it might be something along the lines of displaying knowledge. In the end, I hope that this can help reconsider the relationship between beauty and function as they arise from the friction between society and (formal) technologies, and that it can help shed light on what the realities of writing code are.

So why do this work at all? There are a couple of things which we still do not know about. Since code is a new medium, it's unclear whether it has a specific aesthetic register, or if it borrows from existing registers. Indeed, we will see that it does borrow from different registers (architecture, litterature, science, craft), but it's unclear what are the aesthetic affordances that are specific to source code. An aesthetic lens, and by extension an ethnographic lens, would help in highlighting these issues. There are also very few *examples* of aesthetic manifestations of source code, either because the authors that talk about source code don't always show excerpts of beautiful code, or because the authors that show source code do not connect it back to a broader framework. We don't know if the claims of some people (e.g. code is like language, code is like maths) actually hold up when inspecting the details of code. Another unknown is that, while we do admit that there are multiple kinds of code, it's not clear how all of the different kinds relate to one another. Whether they differ in nature, or in practice, this would contribute to the dependency of an aesthetic register upon its specific medium—to what extent does a programmer have a freedom, or is determined by their technical environment. Speaking of which, technical environments have rarely been taken into account when talking about aesthetics of source code. As such, I would like to know what roles do programming languages play in determining the aesthetic possibilities of a given source code.

And why is this important? For two main reasons, related to fetishism and to our conception of beauty. First, the fetishism of source code is something which can happen when doesn't actually understand the reality of code mught end up ascribing it too much power, or too much influence. Some argue that knowing code is the way to know all of digital media, therefore ascribing abilities to code which might or might not exist, without specific inquiry or verification of what code is, what coders do, and what are they concerned about in the process of writing code. Code might just be too complex to enable the demiurgic possibilities of single individuals through its new and procedural expressiveness. Second, the functional nature of the relationship between source code and software forces us to (re-)think the relationship between aesthetics and efficiency, between beauty and purpose. By looking at what might be the defining medium of the 21st century, we can re-assess our conceptions of aesthetics and beauty in terms of what they achieve, or what they allow to achieve, and therefore reinforcing the cognitive baseline in aesthetic judgments. Additionally, examining the varieties of code can contribute to the discussion on the universality of aesthetics, or its inherent social diversity; are all groups writing code bound to the same set of aesthetic values? By a subset? Or by not set at all? Code as a relatively new artifact allows us to shed new lights on existing debates in aesthetic philosophy.

Finally, why now? In a strictly academic sense, there is a need to connect mutliple strands of thought that are coming to maturation: connecting software studies to programming language design and digital humanities (e.g. Warren Sack) to software ethnographies (e.g. Nick Seaver). This study is part of a more general effort at taking a closer look at code *per se*. Critical code studies analyzes the intent of code, and detect second-order meaning, tying it to the world at large; this study acts as a counterblance and focuses on tying it to its writers and readers—programmers. In a broader sense, it's important because of the place that code is taking in the public debate, about algorithms, big data, etc. As well as professionally: people are learning to be programmers at a very fast rate. So this study could offer a different approach to programming education. That is, providing an additional frame to understand how code can be used both as the main structure of the world and a way to write poems.