

Ugly Code Introduction

GREGOR WEGBERG IS UNHAPPY

Sooner or later every one of us has to write some code for an ETH course. In most cases we get some kind of template which we must use and extend with our own code to get an exercise done. After four years at ETH I am fed up with the quality of those templates.


Introduction

This is going to be a highly subjective article series that revolves around the question “What is beautiful code?”. Of course there is active research going on trying to answer this question. However, I am not going to take the scientific road, but rather use my personal experience to argue about code I perceive as being ugly and how it could be improved. There will not be any definition for “beautiful code” and “ugly code”, although I will try to explain why some code looks ugly to me and why my improved version seems more beautiful.

It is a widely accepted belief by engineers out in the field that code written in the scientific world lacks properties of what is considered to be good-looking code. In a way, this seems intuitive, as most code written by scientists is basically a proof of concept (“POC”). At the same time, in most cases writing code is not their expertise. I do not see anything wrong with that. However, the moment such code is provided to computer science students with the idea that they should learn how to write code, such code quality is not acceptable. How should we learn to write beautiful code if we are provided with such ugly samples? Most of the provided code

I got over the years at ETH would never survive a peer review in any company I have worked for. Therefore, it is time to talk about the situation and provide ideas how to improve such code.

In this article series I showcase code that was provided by an ETH entity to students. I will not tell you who wrote it, although it may be easy to find out. Please note that this is not about the person who wrote the code, but rather how we could learn from such examples to improve our own. It is not about bad-mouthing someone! Be very careful, as I am sure there exists enough very bad-looking code which you yourself wrote. For myself, I know there is enough to find inside my own public repositories.

These articles will be structured in a simple way. First I will showcase some lines of code from one or multiple sources and explain what is wrong with it. Following that, I will try to provide improvements and explain why I think they improve the code quality and therefore make the code better-looking. 

I would be more than happy to get feedback and other opinions. Feel free to write us (visionen@vis.ethz.ch) with positive and negative feedback. If you like, we even could publish it for other students to see that there is no single answer to the problem.