API Design Matters

Michi Henning

In this reading, the author talks about how having good APIs can help programmers significantly, why is it hard to create a good API, and how to write a good API. Good APIs are hard because a small flaw can result in awkward or inefficient code, and humans are prone to errors by nature. For example, a function that overwrites its arguments, does not provide proper return value, and bad variables types can lead to significant cost. The developers either have to write a wrapper function to deal with these flaws, which can lead to more overhead and probably more errors, or fix the problem individually, which will create more complexity in the code. Poor APIs are difficult to understand and make developers take longer time completing a task. Good APIs on the other hand, provides abstractions and makes the code cleaner and more readable. To write an API, one should follow these guidelines: An API must provide sufficient functionality for the caller to achieve its task, be minimal without imposing undue inconvenience on the caller, designed with context and users in mind, should be policy-free for general purpose API and policy-rich for special API, and document should be written before implementation. If APIs are well written with the same semantic/convention, then developers actually learn progressively from functions to functions or even from APIs to APIs. For example, by having the same arguments order, it helps developers save time from consulting documentation.

This reading significantly related to our project as we use Ruby, which supports the use of DRY (Don't Repeat Yourself). Since we will be using a lot of existing codes/APIs, it helps to have good documentation and well defined APIs' culture. Poor written APIs probably is less attractive to use than to rewrite the code that achieves the same task.

I totally agree with the ideas in this reading. I have used several APIs in the past and found that good APIs actually do not require me to look up the documentation as often as bad APIs. Some APIs are very intuitive and it saves me a lot of time developing programs.