Software can be “good” in several ways

- Performance
- Usability
- Readability
- Maintainability / Extensibility
- Potential for reuse

These metrics don’t necessarily go along with each other.

Software sustainability

Long after the project is “done”, it is possible to

- Compile/install/run the software (**Usability**)
- Read and understand the source code (**Readability**)
- Update the program (**Maintainability**)
- Reuse all or part of the program in another project (**Reusability**)

This applies to both the original author(s) and other researchers.

Software as a research output

Authors can reuse their own code, and colleagues can build upon it.

For this to be true, we want software that is - **Readable:** - **Reusable:** - **Extensible:**

Typical hindrance to reusability and extensibility are hard-coded parameters and file paths, global data and lack of modularity.

```
file_handle = open("/Users/Sam/phdthesis/data/survey.dat")
```

Quality software accelerates research

Software that's easier to work with

- Bugs are easier to catch and easier to fix.
- Source code is easy to navigate.
- Source code is easy to modify.

Software that can be built upon

- Code can be understood months or years after development.
- Software can be reused across projects, instead of starting from scratch.
- Software can be extended to new situations.

Software that makes your research more impactful

Open source, quality research software increases credibility and visibility of research.

Hurdles in the way of improving software quality

- Lack of exposure to good examples.
- Too few resources relevant to researchers.
- Lack of training.
- Assumption that the code will never be read.

It’s not as difficult as you might think!

There are simple steps you can take today - Choose descriptive names for variables and functions - Avoid nested logic (for/if/else) - Reduce the size of functions’ parameter lists - Break long functions into smaller functions - Follow a style guide

Code Smells

There are no hard and fast rules about what makes good software. A *code smell* is a pattern of code that **is likely to** make the code hard to understand and work with.

Code smells usually have a name and a clear definition:

- Long Parameter List
- Duplicated Code
- Global Data