Flight Control Laboratory (FCL) Coding Guidelines

Jonas & Tobias

University of Freiburg

jonas.schlagenhauf@imtek.uni-freiburg.de
tobias.schoels@gmail.com

May 23, 2017

Why should I care?

Your only goal:
Produce code efficiently over the *whole* lifetime of your project
$\Rightarrow$ Work as little as possible, as thorough as necessary

Potential benefits:
Reduce maintenance cost (may it be financial or your mental health)

How do I do that?

1. Refactoring tools (auto-formater, auto-indentation,. . .)
2. Documentation (Doxygen, in-line code, external docs)
3. You

1. Refactoring

"*Restructuring code without changing its external behaviour*"

- Many tools nowadays come with your IDE of choice (auto-indentation etc.)
- Documentation tools (Doxygen, in-line code, external docs)

```
/**
 * ... text ...
 */

## Documentation for a method.
#  @param self The object pointer.
```

- Auto-formater (clang-format, autopep8, ...)
- Consistent naming of files, classes, methods, variables (e.g. Camel Case)

2. Documentation

- Header Comments: Explain how to use this piece of code
- Inline Comments: Explain how this code works

Resources:

- `https://google.github.io/styleguide/cppguide.html#Comment_Style`

3. You!

No robot can replace a good coder (... yet)

- Clean Code Development, e.g:
  - Don't repeat yourself
  - Beware of optimization
  - Single responsibilty principle
- Design Patterns, e.g.
  - Builder
  - Singleton

Resources:

`https://sopra.informatik.uni-freiburg.de/soprawiki/CleanCode`
`https://google.github.io/styleguide/cppguide.html`
`https://en.wikipedia.org/wiki/Software_design_pattern`

How do I start?

The good (bad?) thing is, it's up to you:

- Practice
- Read other people's code
- Work in teams