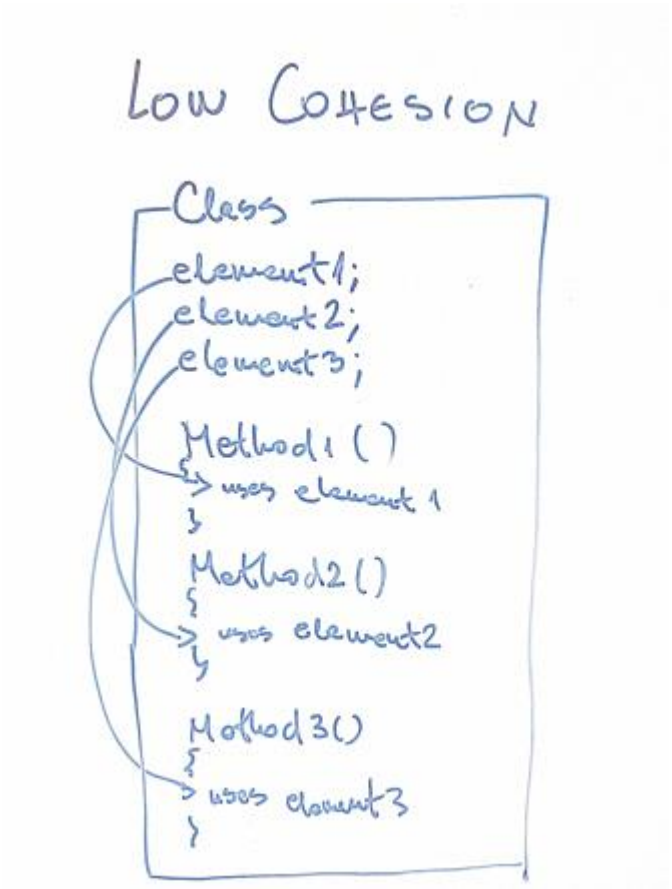
DESIGN PRINCIPLES

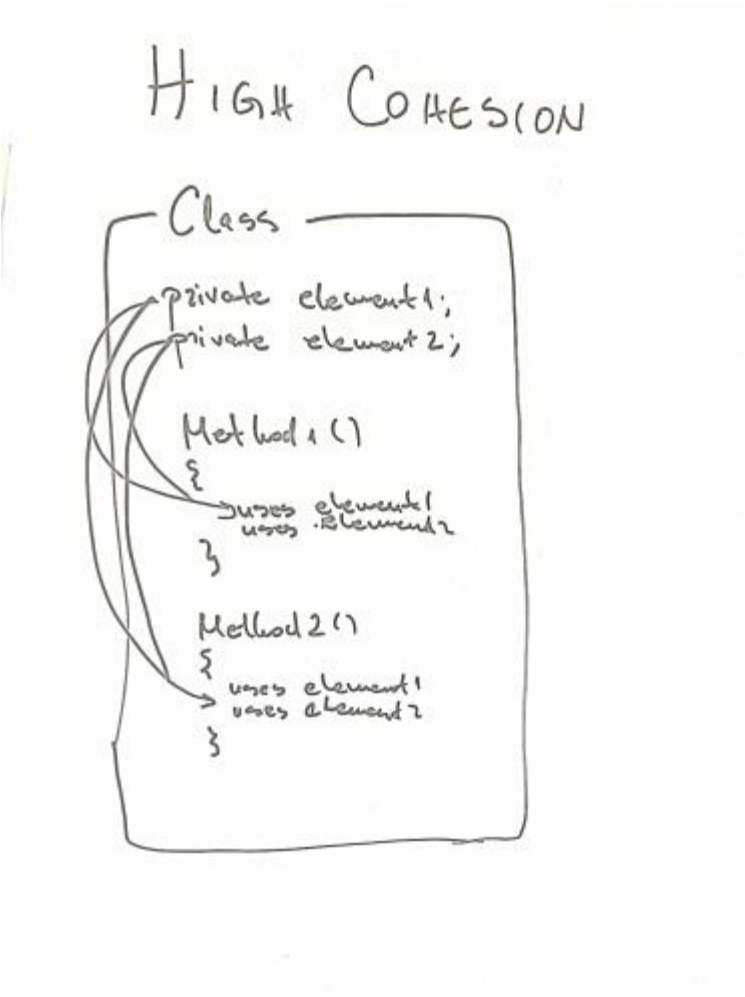
O. DYSHLEVYI


HIGH COHESION
LOW COUPLING

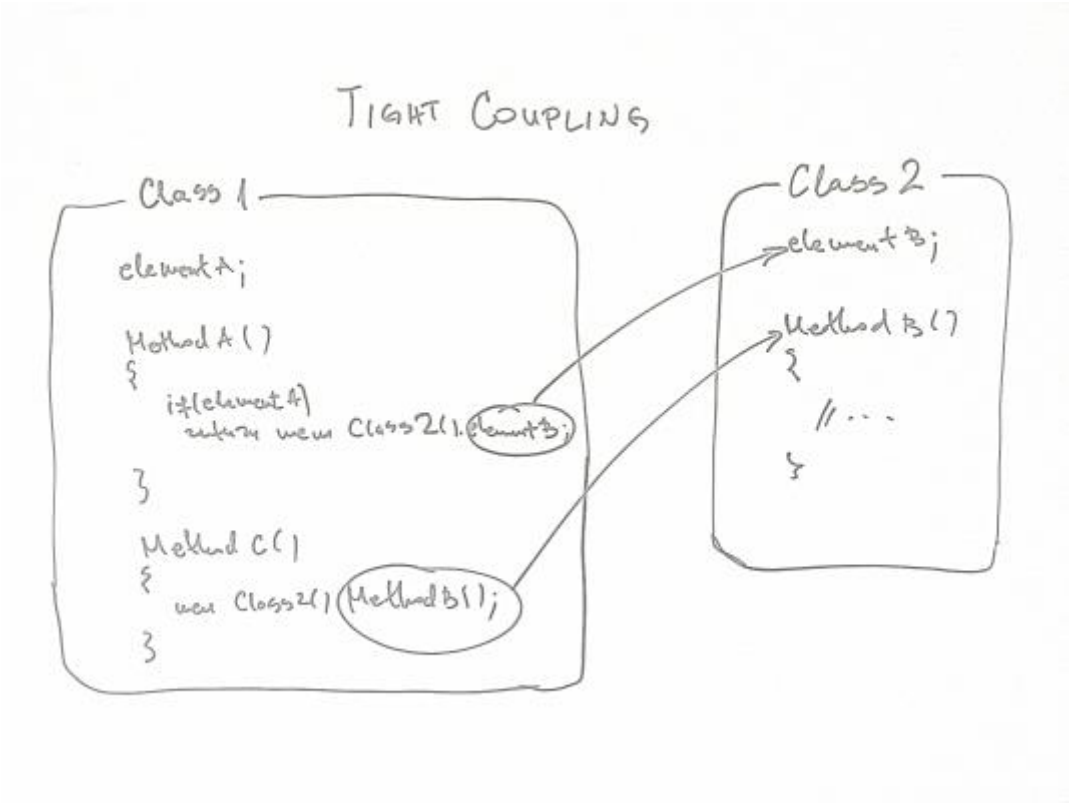
Cohesion	Coupling
Cohesion is the indication of the relationship within module.	Coupling is the indication of the relationships between modules.
Cohesion shows the module's relative functional strength.	Coupling shows the relative independence among the modules.
Cohesion is a degree (quality) to which a component / module focuses on the single thing.	Coupling is a degree to which a component / module is connected to the other modules.
While designing you should strive for high cohesion i.e. a cohesive component/ module focus on a single task (i.e.,single-mindedness) with little interaction with other modules of the system.	While designing you should strive for low coupling i.e. dependency between modules should be less.
Cohesion is the kind of natural extension of data hiding for example, class having all members visible with a package having default visibility.	Making private fields, private methods and non public classes provides loose coupling.
Cohesion is Intra – Module Concept.	Coupling is Inter -Module Concept.

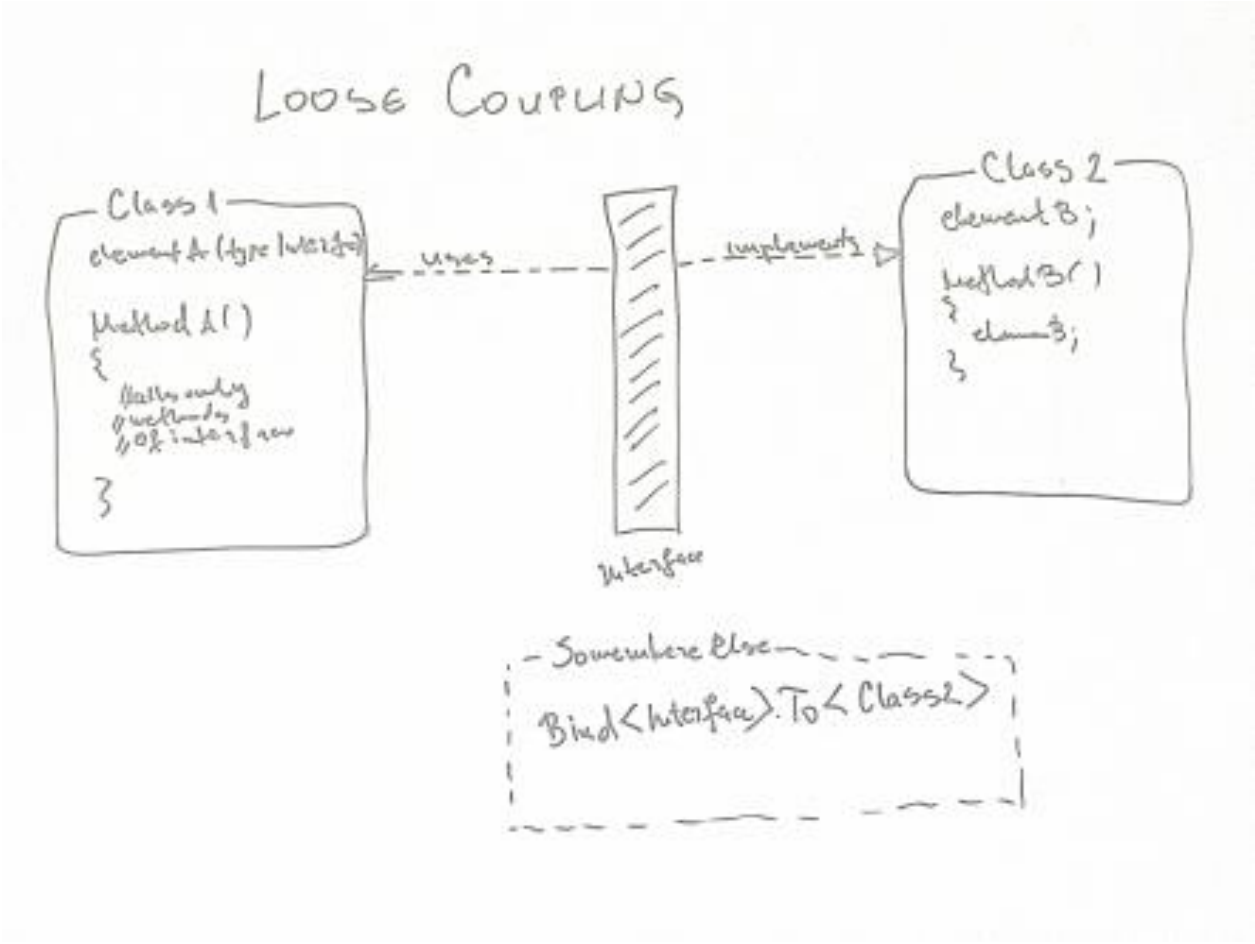
LOOSE COUPLING & HIGH COHESION - COHESION





LOOSE COUPLING & HIGH COHESION - COUPLING





KISS

- Keep it simple stupid
- Prepature optimization is the root of all eveil
- Hence, the "stupid" refers to the relationship between the way things break and the sophistication available to repair them.

- The KISS principle is descriptive to keep the code simple and clear, making it easy to understand
  - Keep your methods small. Each method should never be more than 40-50 lines.
  - Each method should only solve one small problem, not many use cases

VIOLATIONS OF KISS

```
public String weekday1(int day) {
    switch (day) {
        case 1:
            return "Monday";
        case 2:
            return "Tuesday";
        case 3:
            return "Wednesday";
        case 4:
            return "Thursday";
        case 5:
            return "Friday";
        case 6:
            return "Saturday";
        case 7:
            return "Sunday";
        default:
            throw new InvalidOperationException("day must be in range 1 to 7");
    }
}

public String weekday2(int day) {
    if ((day < 1) || (day > 7)) throw new InvalidOperationException("day must be in range 1 to 7");
    string[] days = {
        "Monday",
        "Tuesday",
        "Wednesday",
        "Thursday",
        "Friday",
        "Saturday",
        "Sunday"
    };
    return days[day - 1];
}
```

Any fool can write code that a computer can understand. Good programmers write code that humans can understand.
—Martin Fowler

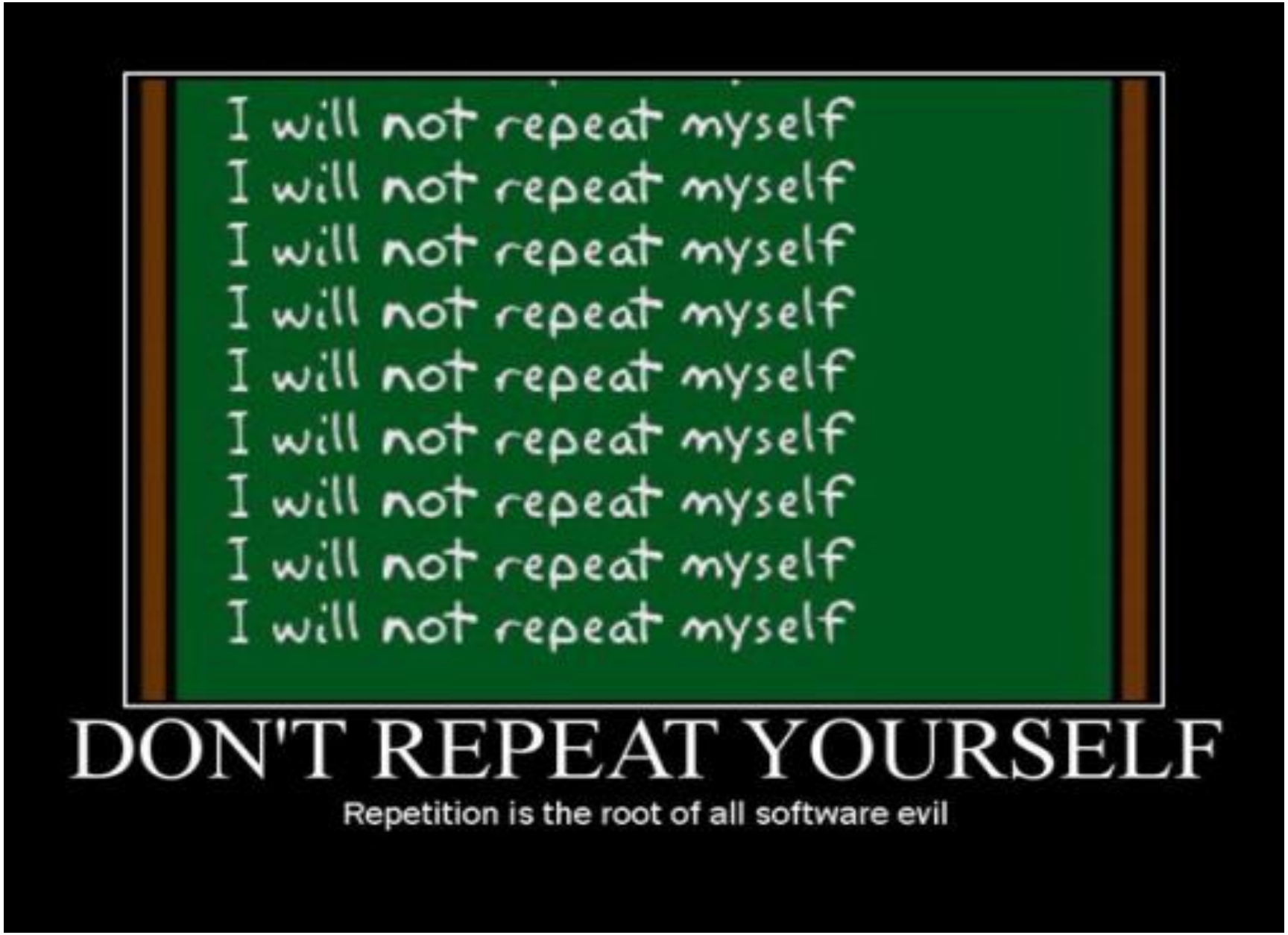
KISS RECOMMENDATIONS

- Keep the scope of variables as small as possible. Don't declare a variable outside an if-block if it's only used within the if-block.
- Don't re-use variables (unless in a loop).
- Keep functions as small as possible and make sure they do just one thing. Having 10 functions doing just one thing is better than 1 function doing 10 things.
- Use proper naming for your variables.
- DRY
- YAGNI
- SOLID

DRY

- reducing repetition of information
- "Every piece of knowledge or logic must have a single, unambiguous representation within a system.“

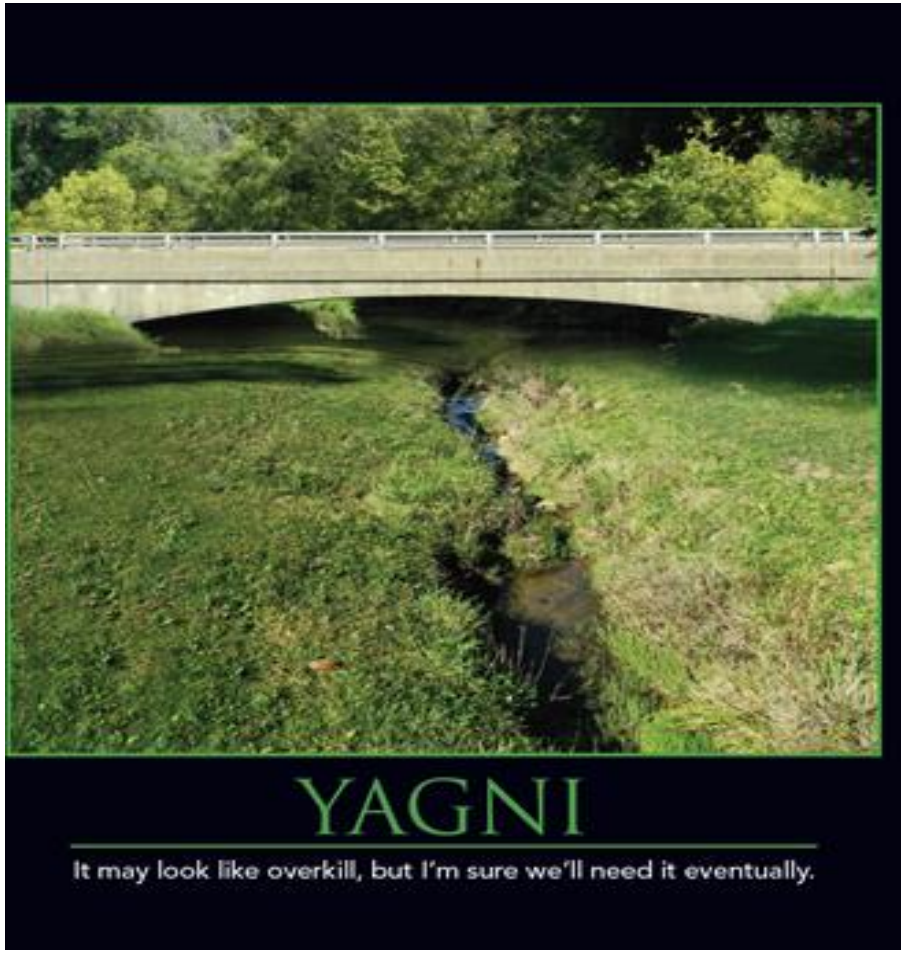
- To avoid violating the DRY principle, divide your code and logic into smaller reusable units and use that code by calling it where you want
- Don't write lengthy methods, but divide logic and try to use the existing piece in your method.


I will not repeat myself
I will not repeat myself
I will not repeat myself
I will not repeat myself
I will not repeat myself
I will not repeat myself
I will not repeat myself
I will not repeat myself
I will not repeat myself
DON'T REPEAT YOURSELF
Repetition is the root of all software evil

YAGNI

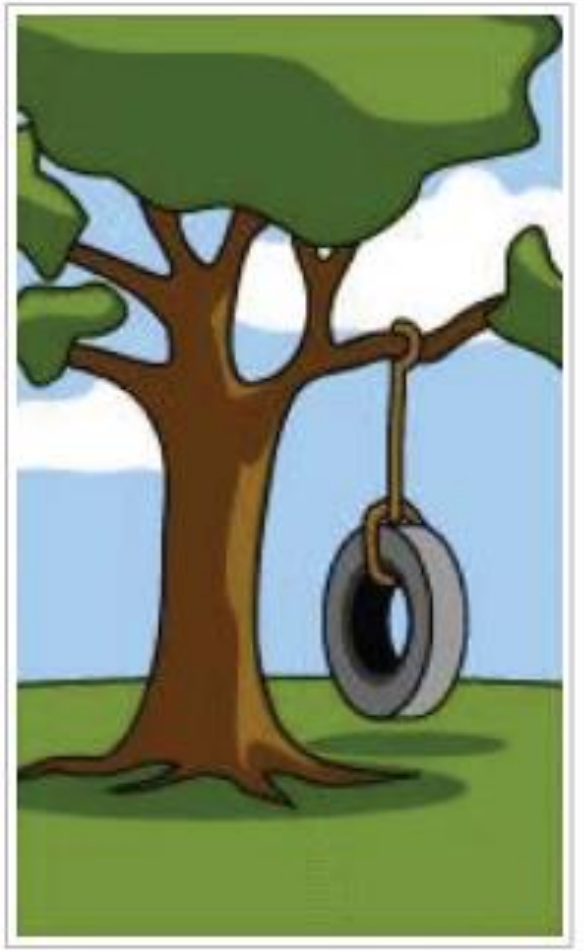
- You ain’t gonna need it
- Contains no more than necessary! It's done when: ~~nothing is missing~~ nothing could be taken away
- The first argument for yagni is that while we may now think we need this presumptive feature, it's likely that we will be wrong

- do not add any functionality until it's deemed necessary



Don’t build this ...

if all you need is this.




STOP
AND
THINK