

# Clean Code okiem Juniora

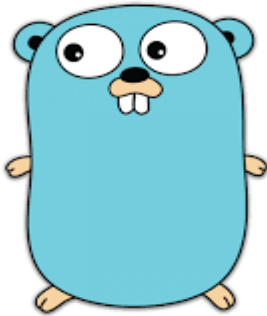
Patryk Zajko



# Agenda

- About me
- What is Clean Code
- Daily Clean Code routine
- Most common mistakes and code improvement techniques
- KISS
- SOLID
- Tools which helps detecting bad quality code and make it easy to fix
- Live Coding
- Q&A
- Summary

## About me





**WRITES UNMAINTAINABLE  
CODE**



”



### DEFINITION OF CLEAN CODE

**WORKS**  
CORRECTLY

**READABLE**  
IT TELLS YOU A STORY

**SIMPLE**  
TO UNDERSTAND

**EFFICIENT**  
IT DOES IT RIGHT

**EXTENDABLE**  
EASY TO MODIFY

**CLEAN**  
SOMEONE CARED

”



### How do I write Clean Code?









# How it ends



© Alamy Stock Photo

\_experience digital transition!

# Best Tools



# What do do!?

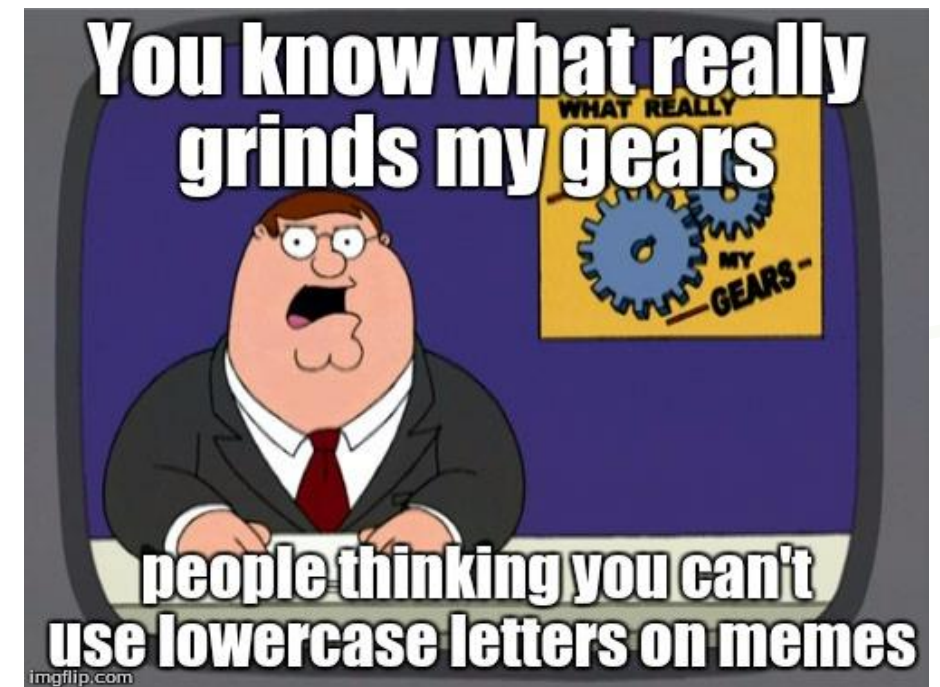


## Let's start from the basics – Meaningful names and functions

- Packages names written lower case

```
package pl.edu.uwb.model;
```

```
public class Student {  
    private static final String UNIVERSITY_NAME = "UwB";  
    private String firstName = "Patryk";  
  
    public String getFirstName() {  
        return firstName;  
    }  
}
```





## Let's start from the basics – Meaningful names and functions

- Class – nouns, names starts with upper case

```
package pl.edu.uwb.model;  
  
public class Student {  
    private static final String UNIVERSITY_NAME = "UwB";  
    private String firstName = "Patryk";  
  
    public String getFirstName() {  
        return firstName;  
    }  
}
```



## Let's start from the basics – Meaningful names and functions

- Methods – verbs, names starts with lower case

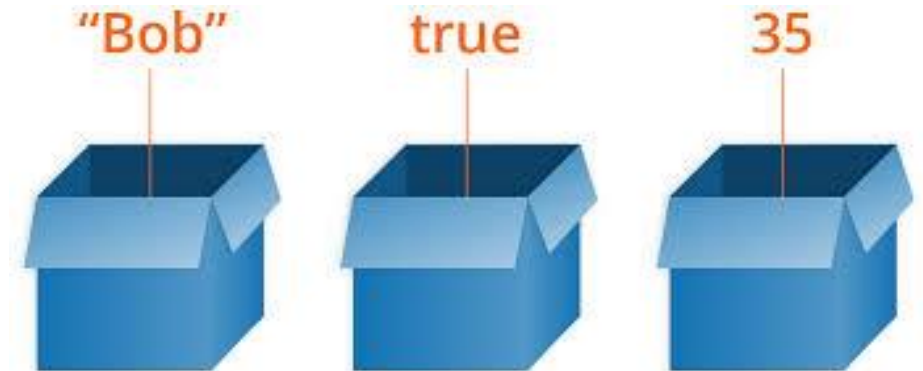
```
package pl.edu.uwb.model;  
  
public class Student {  
    private static final String UNIVERSITY_NAME = "UwB";  
    private String firstName = "Patryk";  
  
    public String getFirstName() {  
        return firstName;  
    }  
}
```



## Let's start from the basics – Meaningful names and functions

- Variables – short but meaningful

```
package pl.edu.uwb.model;  
  
public class Student {  
    private static final String UNIVERSITY_NAME = "UwB";  
    private String firstName = "Patryk";  
  
    public String getFirstName() {  
        return firstName;  
    }  
}
```



## Let's start from the basics – Meaningful names and functions

- CamelCase

```
package pl.edu.uwb.model;  
  
public class Student {  
    private static final String UNIVERSITY_NAME = "UwB";  
    private String firstName = "Patryk";  
  
    public String getFirstName() {  
        return firstName;  
    }  
}
```





## Let's start from the basics – Meaningful names and functions

- Constant fields written upper case

```
package pl.edu.uwb.model;

public class Student {
    private static final String UNIVERSITY_NAME = "UwB";
    private String firstName = "Patryk";

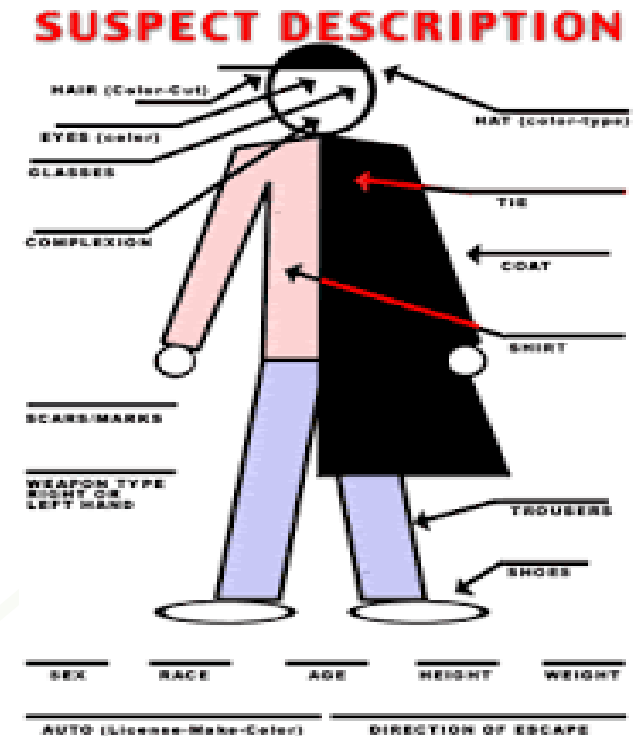
    public String getFirstName() {
        return firstName;
    }
}
```



## Let's start from the basics – Meaningful names and functions

- Self-describing methods and fields

```
package pl.edu.uwb.model;  
  
public class Student {  
    private static final String UNIVERSITY_NAME = "UwB";  
    private String firstName = "Patryk";  
  
    public String getFirstName() {  
        return firstName;  
    }  
}
```





isTheMethodReturnsTheCorrectValueOr  
IsAThanosWhichWillConquerTheWorld  
AndIfThePersonHasTheCorrectPasswor  
dAndIfNoThePasswordIsChanged(int x,  
int y, String b)

isOk(Mother yours)

## Method names pattern

Prefix	Short description
get	Gets something from an object
set	Setting a property
is	Checks if something is true or false
check	Checks if something is true, throwing an exception if it is not true
contains	Checks if a collection contains the queried object
create	Creates object
test	Tests some functionalities





# Loggers

```
if (studentList.isEmpty()) {  
    //System.out.println("Student list is empty!");  
    LOGGER.info("Student list is empty!");  
}
```

Logging Level	Description
ALL	Includes all leveles of logging
DEBUG	Useful to debug an application
ERROR	Used when error occurs and application might continue running
FATAL	Used when very severe error occurs which might lead to application crash
INFO	Used for informational purposes, showing information what is the progression of our appliaction
OFF	Highest possible level, used to turn off logging
TRACE	Same as debug
WARN	Used in case of harmfull condition

```
private Collection<Object[]> parametersForStudentType() {  
    return new ParametersCsvReader<Object[]>(this.getClass()  
        .getResourceAsStream(STUDENT_TYPE_DEFINITION_PARAMETERS_FILE))  
        .map(r -> new Object[]{r.get(1), r.get(2), r.get(3), r.get(4)})  
        .read();  
}
```



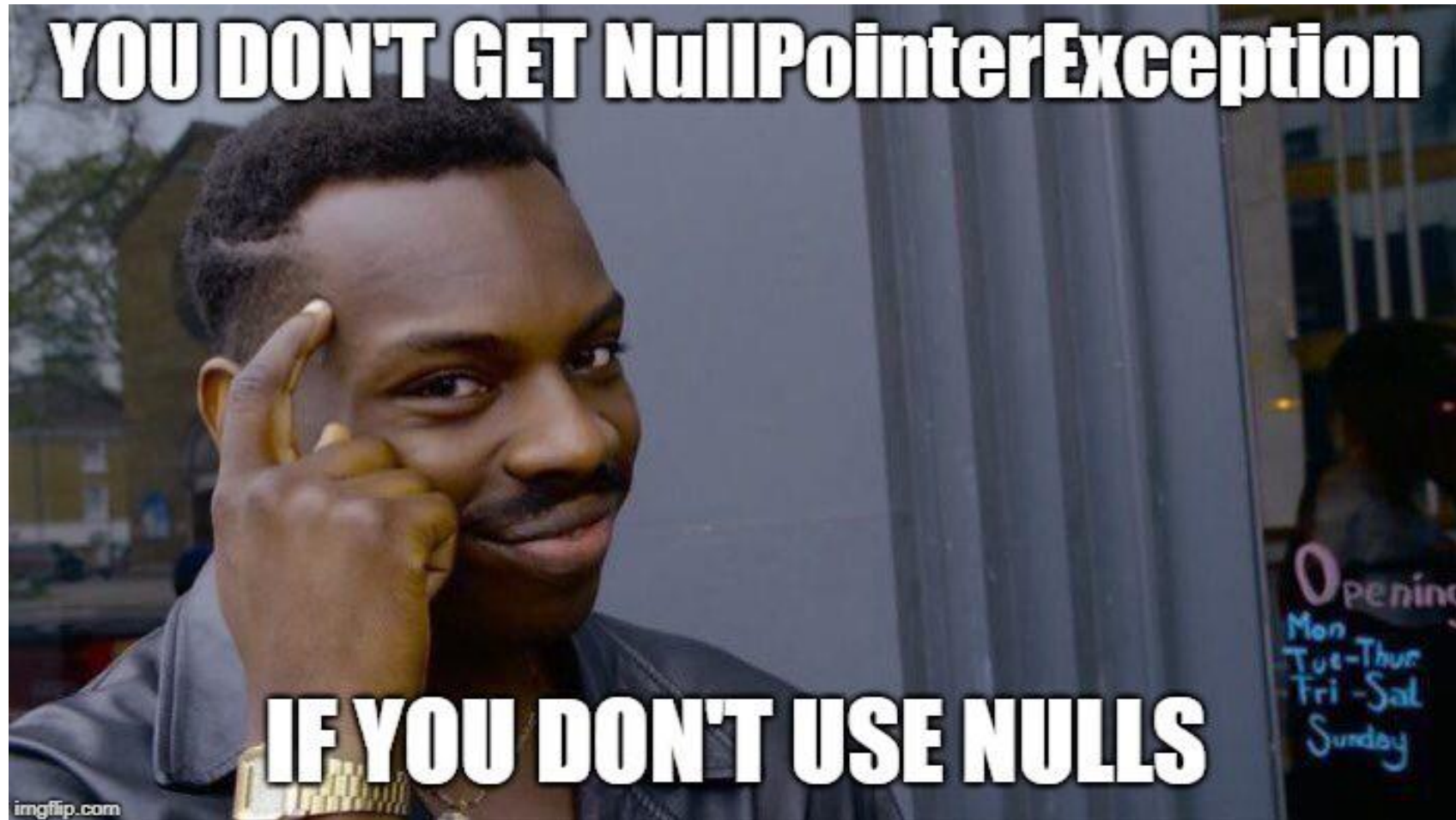
Magical powers.

```
private Collection<Object[]> parametersForStudentType() {  
    return new ParametersCsvReader<Object[]>(this.getClass().  
        getResourceAsStream(STUDENT_TYPE_DEFINITION_PARAMETERS_FILE))  
        .map(r -> new Object[]{r.get(FIRST_NAME), r.get(LAST_NAME),  
            r.get(AGE), r.get(GENDER)})  
        .read();  
}
```





## Return null?



```
private String safeString(String string) {  
    if(string != null) {  
        return string;  
    } else {  
        return null;  
    }  
}
```

Not sure if this string is safe enough

# Method Parameters



```
public Note(String a, String b, boolean c) {  
    this.name = a;  
    this.description = b;  
    this.isCompleted = c;  
}
```



```
public Note(String name, String description, boolean isCompleted) {  
    this.name = name;  
    this.description = description;  
    this.isCompleted = isCompleted;  
}
```

When I wrote this code,  
only God & I understood what it did.



Now...  
only God knows.

# Comments

```
//Comments single line
```

```
/*  
Comments multi lines
```

```
-  
-  
-  
*/
```

```
/**  
 * Documentation comments  
 * @param name  
 * @param description  
 * @param isCompleted  
 * @return  
 */
```

Comment type	Description
Single line	Easiest type of comments, describing the code functionality
Multi lines	To overcome „//” symbol each line better use multi lines comment
Documentation comments	Used generally when writing code for a bigger project or open source.

KISS!

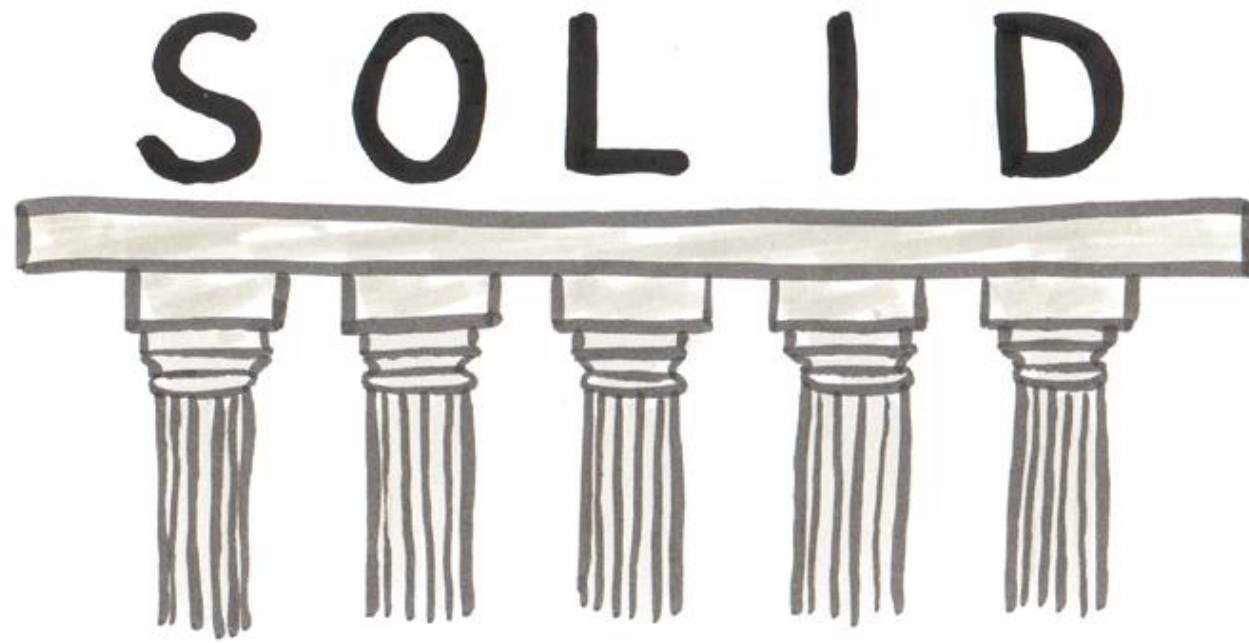


K.I.S.S.

Keep It Simple, Stupid!







- **S** – Single responsibility principle

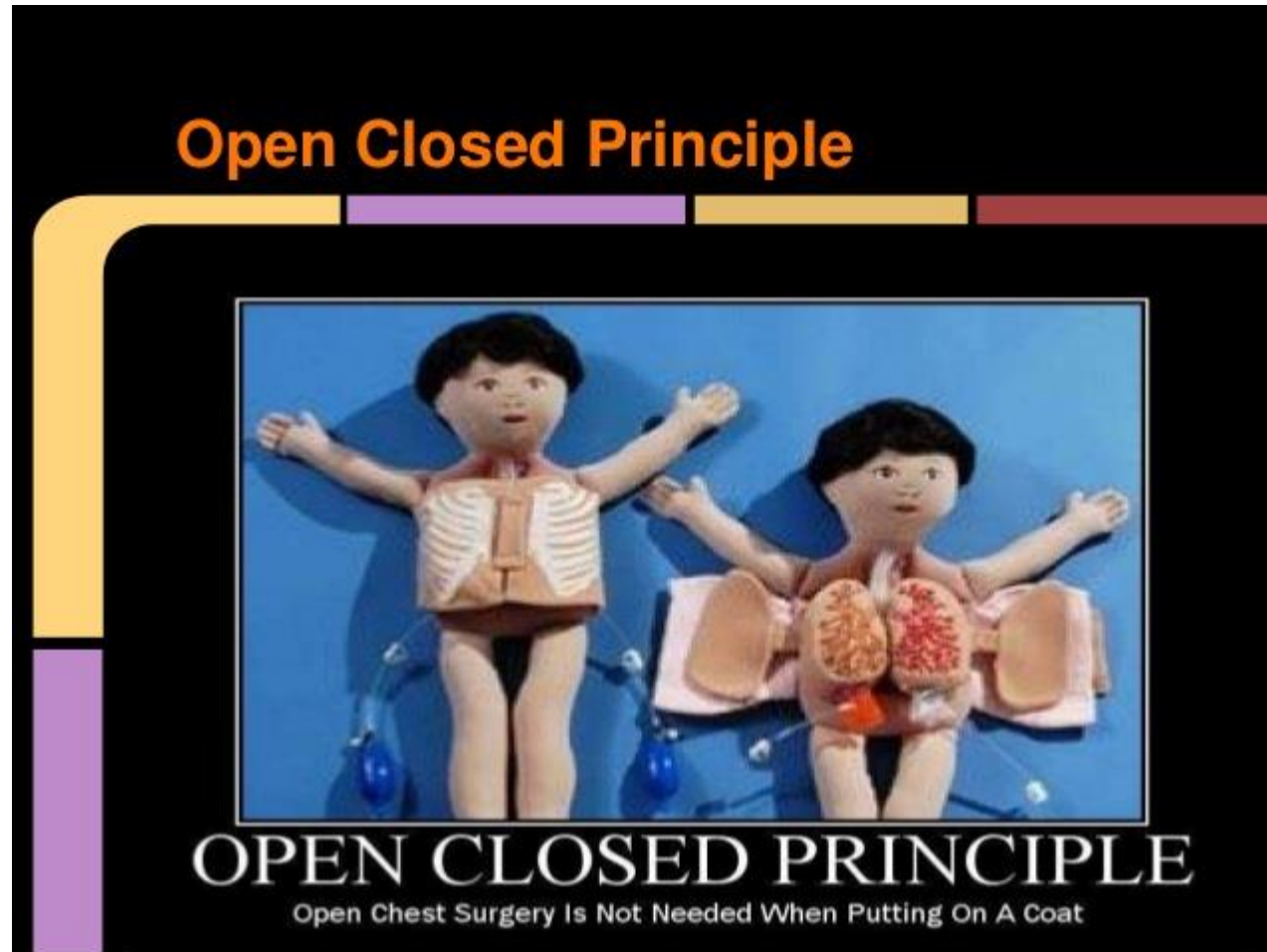


## Single Responsibility Principle

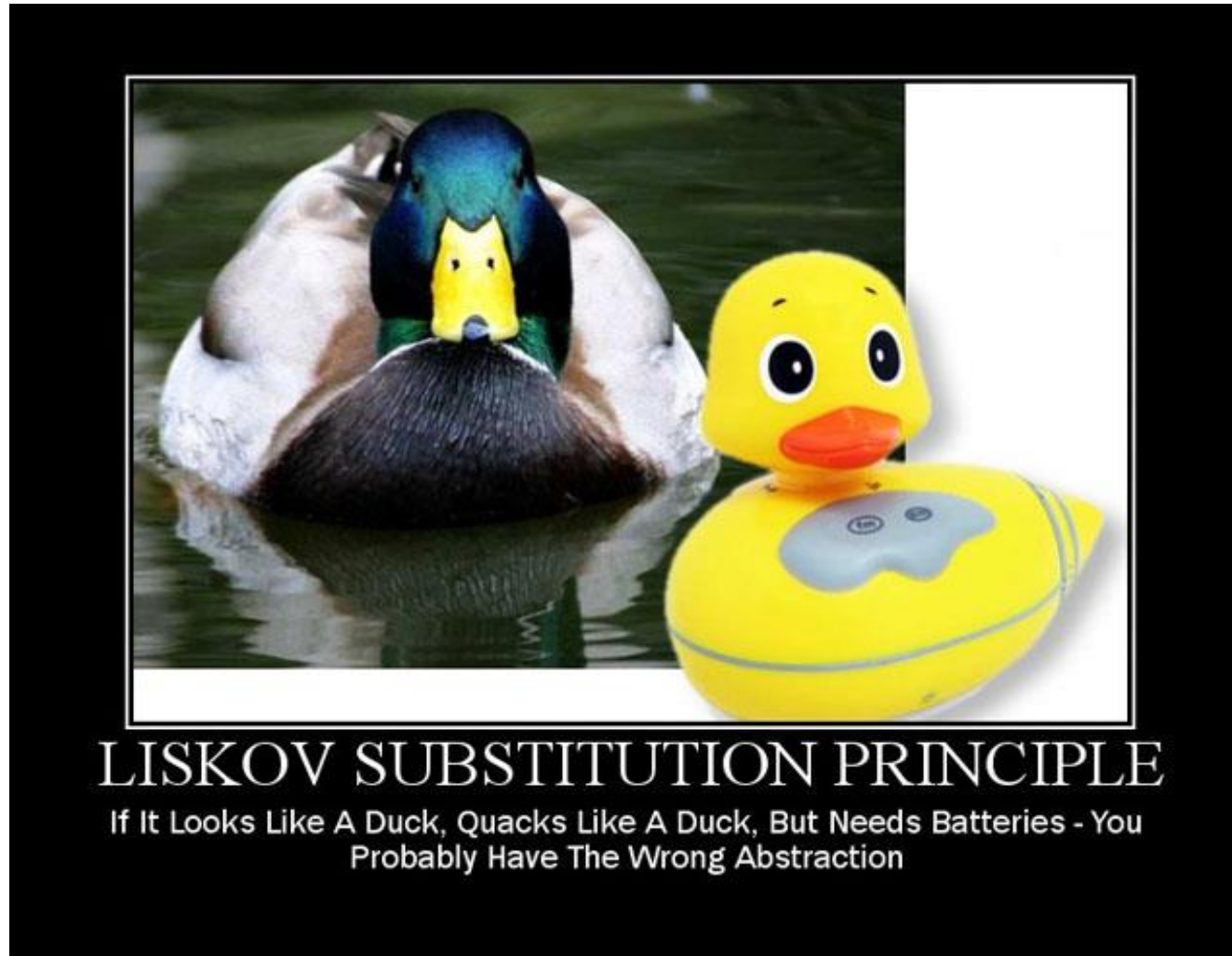
**Just because you *can* doesn't mean you *should*.**

#learn2code Why don't programmers on the left understand that increasing the power and scope of the federal government breaks this SOLID principle?

- ○ – Open/closed principle



- L – Liskov substitution principle

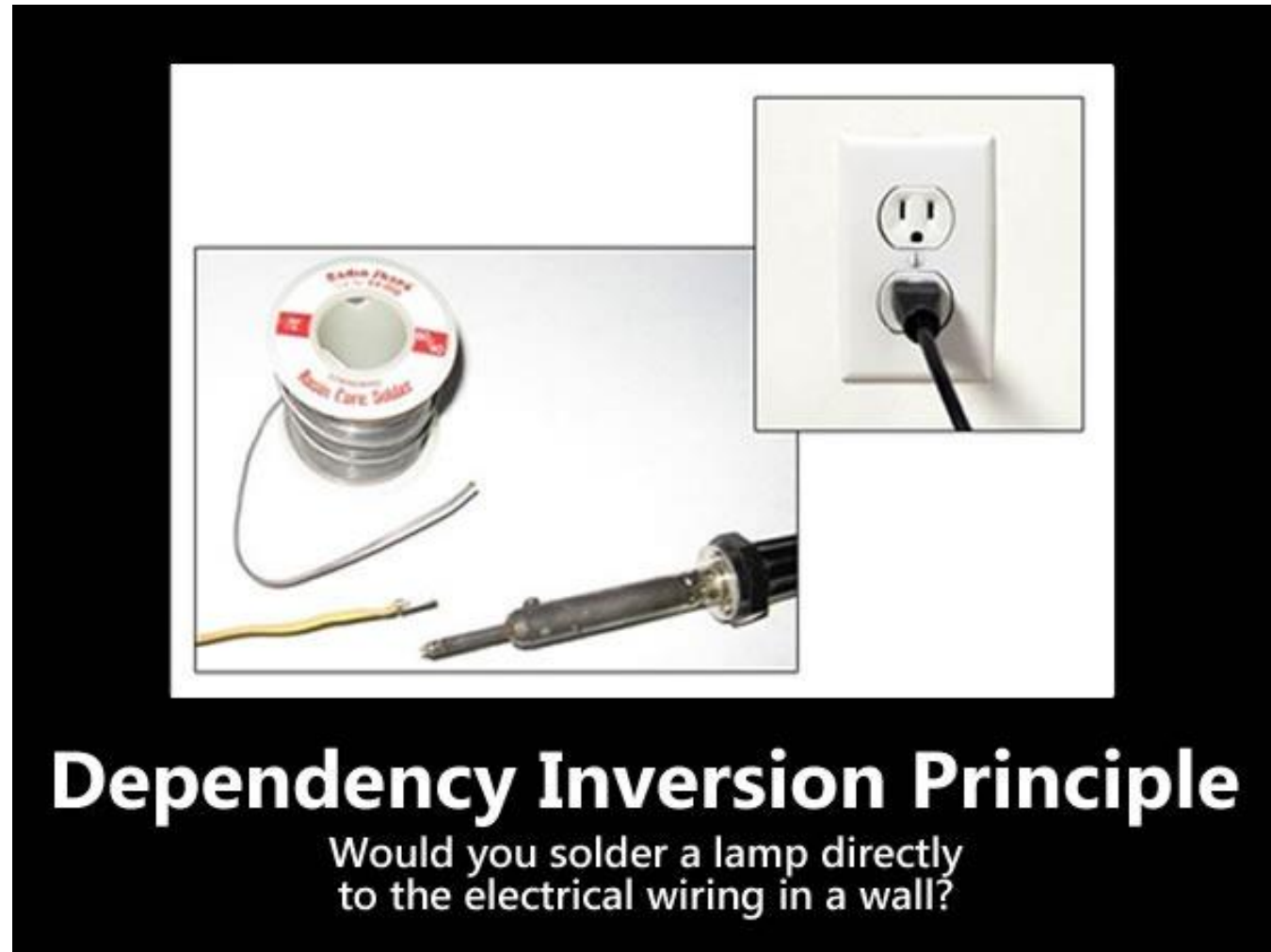


- | – Interface segregation principle





- D – Dependency inversion principle



## Code smell & refactor – best tools to make our code cleaner

sonarqube



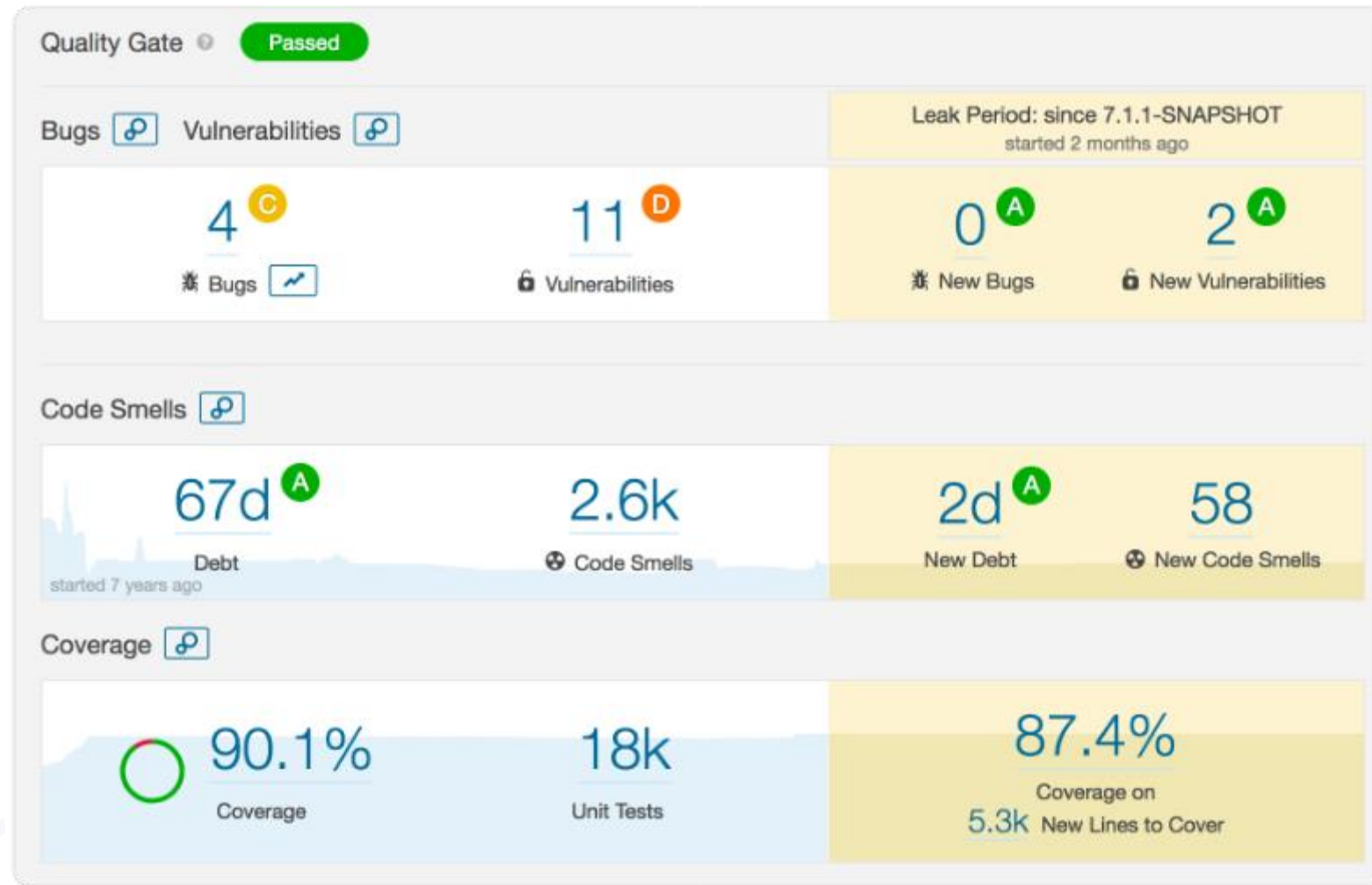
sonarlint

**"I will refactor that  
code later today"**

And  
Other Hilarious Jokes  
You Can Tell Yourself

Volume II

My favourite programming  
book.





## Clean Code Summary

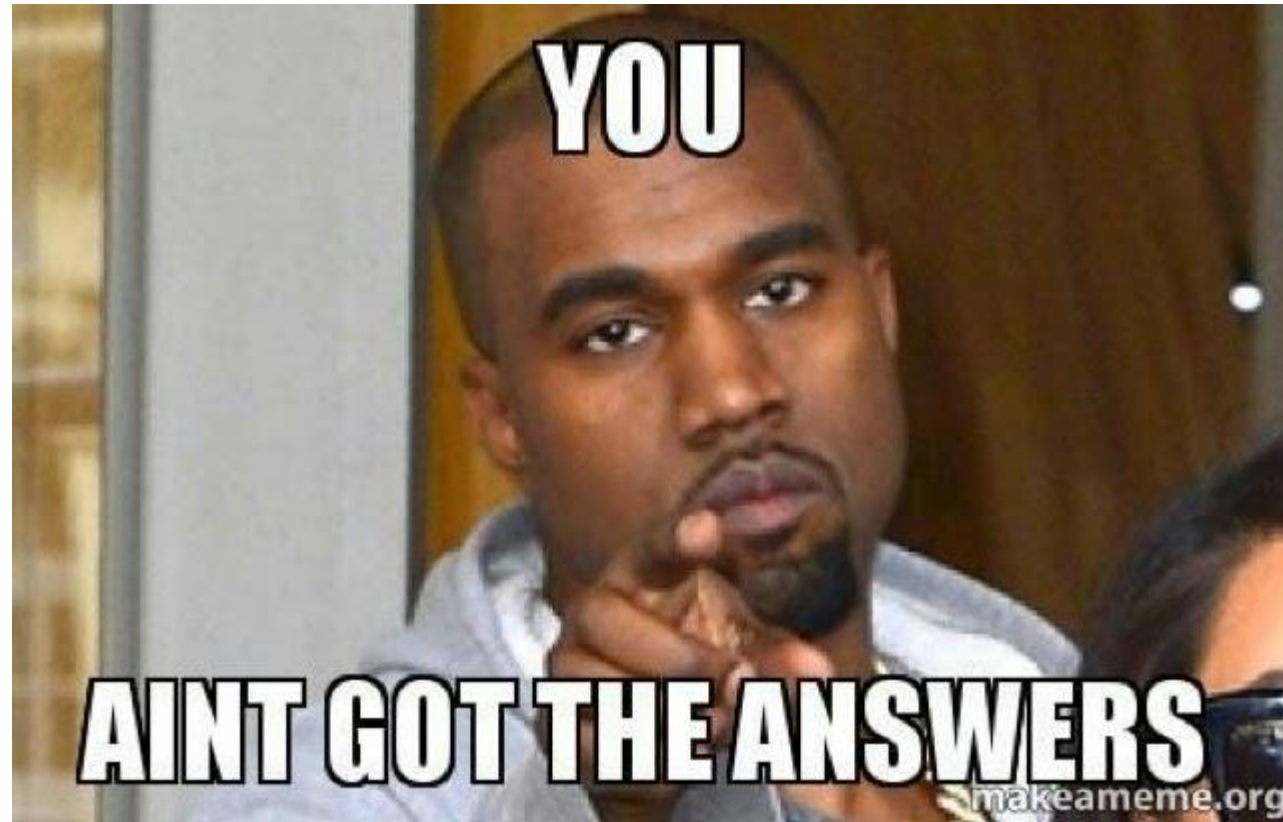
1. Theoretical knowledge is basic
2. Look for open source repositories
3. Use code smell tools
4. Don't get obsession over Clean Code – one dirty line won't kill you
5. Writing code is like writing a story, let somebody else read it

The most important one – have fun while coding😊












Q&A







PROCEDURE						
						
Drain the washer fluid tank	Take 15mm Wrench	Unmount the washer fluid tank	Remove the washer fluid tank	Inspect the washer fluid tank	Replace the washer fluid tank	Refill the system
PART NUMBER	PART NUMBER	PART NUMBER	PART NUMBER	PART NUMBER	PART NUMBER	PART NUMBER
0000001245	0000001255	0000001273	0000001273	0000001273	0000001274	0000001274



## Praktyki Letnie oraz praca MPJ Transition Technologies PSC





## PRAKTYKANT/ PROGRAMISTA

### MIEJSCE PRACY

Białystok

### NUMER REFERENCYJNY

PP/Białystok\_2019-03-27\_1541

### DATA APLIKACJI

2019-06-30

**APLIKUJ**



## OFERUJEMY:

- 3 miesięczne (lipiec-wrzesień), płatne praktyki w wymiarze 4/5 etatu (32 godz. tygodniowo)
- Ciekawe i pełne wyzwań projekty w najnowocześniejszych technologiach
- Oferty pracy dla najlepszych i najbardziej zaangażowanych

## ZAKRES OBOWIĄZKÓW:

- Zapoznanie się z procesem wytwarzania oprogramowania
- Współtworzenie aplikacji
- Projektowanie, implementacja i testowanie kodu
- Wytwarzanie oprogramowania w oparciu o metodologię SCRUM

## WYMAGANIA:

Jeśli:

- Chcesz poznać lub już znasz: Java, Scrum, GIT, Jenkins, Eclipse, AngularJS, Spring, C# (Hololens), Angular, aplikacje mobilne
- Nie są Ci obce aspekty programowania obiektowego
- Świadomie wykorzystujesz wzorce projektowe
- Swobodnie komunikujesz się w języku angielskim
- Chciałbyś zdobyć doświadczenie i poznać jak wygląda praca w IT
- Chcesz wziąć udział w projektach podążających za najnowszymi trendami w IT
- Chciałbyś uczyć się od najlepszych

Ta oferta praktyk jest dla Ciebie !!!



- Java
- Spring
- Hibernate
- Lombok
- H2
- AWS SDK
- Angular 6



### Get started!

Explore Machine Learning in your life and use algorithm to change the way you used to work. With this simple app you can predict when your engine, batteries, or some other device will fail. You don't need expensive tech or nerd squad, all things will have been calculated in Amazon Web Service, so you receive results as fast as possible.

[Sign up](#)[Log in](#)

Machine Learning

New modelModelsFiles

[« Go to models list](#)

Create model

1 Choose file

2 Schema

3 Add name

4 Done

Create on new fileCreate on old file

Start over

Activate Windows  
Go to Settings to activate Windows.

Machine Learning

New modelModelsFiles

## Your models

Create new model

Refresh model list

Name	Status	
Local computation - model: Model number one	COMPLETED	Details
Local computation - model: Kurnik o zaostrzonym rygorze	COMPLETED	Details
Local computation - model: Model on old file	COMPLETED	Details
Local computation - model: My model	COMPLETED	Details

[« Go to models list](#)[Create new evaluation](#)[Create new prediction](#)[Model details](#)[Evaluations](#)[Predictions](#)

Local computation - Multiple on old file

[Click to see prediction details](#)

Local computation - Single on new file

[Click to see prediction details](#)**Id:** bp-LajL2GNQs9q**Data Source Id:** ds-FkvXQXLhs5y**Status:** COMPLETED[Show results](#)[Delete prediction](#)

Local computation - Multiple on old file

[Click to see prediction details](#)

Local computation - Multiple observations - new file

[Click to see prediction details](#)

Activate Windows

Go to Settings to activate Windows.



## MŁODSZY PROGRAMISTA JAVA

### MIEJSCE PRACY

Białystok

### NUMER REFERENCYJNY

MPJ/Białystok\_2014-04-25\_1047

### DATA APLIKACJI

2019-12-31

**APLIKUJ**





### OFERUJEMY:

- Rozwojową pracę z najnowszymi technologiami informatycznymi
- Zaawansowany i intensywny program szkoleń zewnętrznych i wewnętrznych
- Możliwość udziału w jednej/wielu z technologicznych grup community działających w firmie w celu propagacji wiedzy
- Pracę w międzynarodowych, doświadczonych zespołach dla największych światowych korporacji przemysłowych
- Bogaty pakiet socjalny - wczasy pod gruszą, paczki świąteczne, karnety na ośrodki sportowo-rekreacyjne, słodkie i owocowe dni
- Elastyczny czas pracy dla osób studiujących
- Kursy językowe w biurze
- Młody dynamiczny zespół
- Niepowtarzalną atmosferę pracy stabilizację

### ZAKRES OBOWIĄZKÓW:

- Analiza wymagań klienta
- Implementacja rozwiązań biznesowych w oparciu o dokumentację techniczną
- Dokumentowanie kodu
- Dbanie o jakość kodu i zgodność ze standardami obowiązującymi w firmie
- Integrowanie rozwiązań z istniejącymi systemami

## WYMAGANIA:

- Znajomość Java
- Znajomość SQL/ JSP / HTML / CSS / JS / XML
- Chęci i umiejętności szybkiego uczenia się
- Znajomość języka angielskiego

## MILE WIDZIANE:

- Doświadczenie na podobnym stanowisku
- Znajomość technologii Spring, Hibernate
- Znajomość języka niemieckiego, francuskiego bądź włoskiego
- Gotowość do wyjazdów zagranicznych