

Questionnaire

1 Welcome Page

Welcome to the Clean Code survey

The survey has been developed by a bachelor thesis student and a researcher at Blekinge Tekniska Högskola, Kevin Ljung and Javier Gonzalez Huerta.

This study aims at understanding how clean code principles are perceived and used in practice. In a nutshell: what do we do to write clean code?

If you have further questions, please do not hesitate to contact us: kevinljung98@gmail.com or jgh@bth.se

Thanks for your participation!

Kevin and Javier

2 Informed Consent

Informed Consent Form

The purpose of this form is to inform you about the Clean Code in Practice project and the details of your participation in this study.

1. Who is running this study?

The study is conducted by Kevin Ljung, a bachelor thesis student, and Javier Gonzalez Huerta, a researcher at the Department of Software Engineering (DIPT) at Blekinge Institute of Technology.

2 What is the purpose of this study?

The study aims at identifying the essential clean code practices and principles that the developers find helpful in practice and know whether developers write clean code initially or messy code that needs to be refactored later. We also aim to investigate whether the developers believe that clean code eases reading, understanding, reusing, or modification.

3 What data are we collecting? Is my identity as a participant safe?

No personal data is collected, and therefore no data can make you, as a respondent, identifiable. We will only analyze aggregated data, protecting in that way the anonymity of every participant. We only collect data related to clean-code practices, how to write clean-code etc. We will only use the questions about age group or gender for describing the demographics of the study and are not going to be used in the analysis.

4. Who will have access to my responses and data?

Only the student and the researcher referred to as "us" or "we" will have access to the responses and data. However, the anonymized answers will be analyzed and stored until the analysis is complete.

Then we will destroy all the raw data, and we will only work with the aggregated data.

The results from the analysis might be published in scientific venues, and we will ensure not to include any information that might make you directly identifiable in any way. We will handle all data following strict ethical practices.

We might store the anonymized data in shared documents for its analysis, assuring that only we have access to the raw data. The files will not be available to anyone else outside of the study.

5. Do I have to do this?

Your participation is entirely voluntary. You can withdraw from the study at any point, and your data will be not be stored and therefore not analyzed.

6. Can I ask questions?

We would like to hear about your experience of your participation in this survey after completing it.

You can contact us by email to answer any final questions you might have and for us to clarify anything that wasn't clear. Drop us an email, and we will be happy to answer questions.

7 Keeping participants well and healthy

It is very unlikely that participating in this study will trigger any unpleasant thoughts or feelings. However, if this does occur, we advise you to contact us as soon as possible.

- ☐ I confirm that I have been informed about how data is going to be used, and I therefore volunteer to participate

3 Demographics

To which gender identity you most identify?

Please fill in the gender identity you most identify with.

- ☐ Male
☐ Female
☐ Other

What age group do you belong to?

- ☐ < 18
☐ 18 - 25
☐ 26 - 30
☐ 31 - 40
☐ 41 - 50
☐ 51 - 60
☐ 60+

What is your highest education degree?

(Optional)

Please fill in the highest education you have completed.

- ☐ No higher education degree
☐ Bachelor degree (up to 3 years of university education)
☐ Bachelor + MSc degree - or equivalent- (up to 5 years of university education)
☐ PhD
☐ Other

How many years of programming experience do you have?

(Optional)

- ☐ < 1 year

- ☐ 1 - 3 years
- ☐ 4 - 6 years
- ☐ 7 - 9 years
- ☐ 10 - 20 years
- ☐ 20+ years

I am comfortable programming in [language]

(Optional)

Fill in how comfortable you are with the programming language using the corresponding Likert scale.

	Strongly Disagree	Disagree	Somewhat Disagree	Neither Agree nor Disagree	Somewhat agree	Agree	Strongly Agree
Python	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
JavaScript	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Java	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
C++	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
C	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
C#	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Kotlin	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Swift	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Go	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Scala	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Other (also write which one)	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
<input type="text"/>							

What area do you work within?

(Optional)

What type of company do you work for? Or which type of systems do you usually develop?
Just leave it blank if you do not want to answer or cannot answer this question.

4 RQ1

This survey is about Clean Code. When you write your answers about Clean Code: What programming language do you have in mind?

Some of the answers might vary depending on the programming language. It is important for us to know in which programming language you think when you answer the clean-code specific questions.

- ☐ Python
- ☐ JavaScript
- ☐ Java
- ☐ C++
- ☐ C
- ☐ C#
- ☐ Kotlin
- ☐ Swift
- ☐ Go
- ☐ Scala

☐ Other (Write which one)

I have heard of some of the essential Clean Code practices and principles before.

- ☐ Strongly disagree

- ☐ Disagree
☐ Somewhat disagree
☐ Neither Agree nor Disagree
☐ Somewhat agree
☐ Agree
☐ Strongly Agree

General principles

Do you agree with the clean-code [principle]

Fill in how much you agree by using the corresponding Likert scale.

	Strongly Disagree	Disagree	Somewhat Disagree	Neither Agree nor Disagree	Somewhat agree	Agree	Strongly Agree
The Boy Scout Rule (more info)							
Leave the code cleaner than you found it	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Minimize nesting (more info)							
Avoid nesting if-statements, for-loops, etc. when possible	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
KISS - Keep It Simple, Stupid! (more info)							
Keep the code simple	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
OCP - Open Closed Principle (more info)							
This principle means that a function, module, class, etc., should be extensible but not open to modification [1].	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Separate Constructing a System from Using It (more info)							
Separate the logic for creating objects and for using objects. The logic to create the objects should not be intermixed with the logic that uses the objects.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

Naming principles

Do you agree with the clean-code [principle]

Fill in how much you agree by using the corresponding Likert scale.

	Strongly Disagree	Disagree	Somewhat Disagree	Neither Agree nor Disagree	Somewhat agree	Agree	Strongly Agree
Use Meaningful Names (more info)							

Names should mean something within the code to the developers

☐ ☐ ☐ ☐ ☐ ☐ ☐

Use Intention-Revealing Names ([more info](#))

The name of a variable, function, or class should reveal the intent, what it does or how it is used.

☐ ☐ ☐ ☐ ☐ ☐ ☐

Pronounceable Names ([more info](#))

A name should be easy to pronounce orally

☐ ☐ ☐ ☐ ☐ ☐ ☐

Searchable Names ([more info](#))

To find the name when searching for it within the code

☐ ☐ ☐ ☐ ☐ ☐ ☐

Avoid Disinformation ([more info](#))

Do not confuse programmers with false names, that obscures what the code does

☐ ☐ ☐ ☐ ☐ ☐ ☐

Avoid Mental Mapping ([more info](#))

Developers should not have to map a concept that they already know by another name

☐ ☐ ☐ ☐ ☐ ☐ ☐

Function and Method principles

Do you agree with the clean-code [principle]

Fill in how much you agree by using the corresponding Likert scale.

	Strongly Disagree	Disagree	Somewhat Disagree	Neither Agree nor Disagree	Somewhat agree	Agree	Strongly Agree
Do One Thing (more info)							
Do one thing, and not multiple things	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Command Query Separation (more info)							
Functions are supposed to do something or answer something, but they should not do both [1].	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Extract Try-Catch Block (more info)							
Extract Try-Catch block into a function on its own.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Have No Side Effects (more info)							
A side effect is when a function is supposed to do only one thing and	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

does other things.

DRY - Don't Repeat Yourself ([more info](#))

No duplication or alike (e.g. almost identical blocks of code)

☐ ☐ ☐ ☐ ☐ ☐ ☐ ☐

Function Arguments ([more info](#))

Only 1 - 3 arguments passed to a function

☐ ☐ ☐ ☐ ☐ ☐ ☐ ☐

Structured Programming ([more info](#))

Large functions should have one entry, only one exit (return). Avoid *break*, *continue*, and *goto*

☐ ☐ ☐ ☐ ☐ ☐ ☐ ☐

Methods/Functions should be small ([more info](#))

The size/length of a function or method should be small

☐ ☐ ☐ ☐ ☐ ☐ ☐ ☐

Comment principles

Do you agree with the clean-code [principle]

Fill in how much you agree by using the corresponding Likert scale.

	Strongly Disagree	Disagree	Somewhat Disagree	Neither Agree nor Disagree	Somewhat agree	Agree	Strongly Agree
Amplification (more info)							
Explain why a certain change is very important	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Clarification (more info)							
If it is unclear what the code does, then use a comment to explain what it does	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Explain Yourself in Code (more info)							
Try to explain in code without using comments if possible	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Explanation of Intent (more info)							
When the intent is not self-explained by the code, write a comment.		<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	
TODO Comments (more info)							
Leave TODO comments when you think something should be done, but	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

you cannot implement it at the moment.

Warning of Consequences ([more info](#))

Warn other developers about running the code for some reason

☐ ☐ ☐ ☐ ☐ ☐ ☐

Formatting principles

Do you agree with the clean-code [principle]

Fill in how much you agree by using the corresponding Likert scale.

Strongly Disagree Disagree Somewhat Disagree Neither Agree nor Disagree Somewhat agree Agree Strongly Agree

Team Coding Standards ([more info](#))

Reach a consensus within the team about a common coding standard to use

☐ ☐ ☐ ☐ ☐ ☐ ☐

Horizontal Formatting - Indentation ([more info](#))

X-axis, left to right. Indentation is achieved with tabs or spaces.

☐ ☐ ☐ ☐ ☐ ☐ ☐

Dependent Functions ([more info](#))

"The caller should be above the callee, if at all possible." [1] (Vertically means Y-axis, top to bottom).

☐ ☐ ☐ ☐ ☐ ☐ ☐

Vertical Distance and Ordering ([more info](#))

Blank lines, close related lines, ordering of lines,

☐ ☐ ☐ ☐ ☐ ☐ ☐

Organizing for Change ([more info](#))

Organize so that classes are not so sensitive to change. Change should not make the code break.

☐ ☐ ☐ ☐ ☐ ☐ ☐

Object and Data Structure principles

Do you agree with the clean-code [principle]

Fill in how much you agree by using the corresponding Likert scale.

Strongly Disagree Disagree Somewhat Disagree Neither Agree nor Disagree Somewhat agree Agree Strongly Agree

Data/Object Anti-Symmetry ([more info](#))

Objects hide the implementations of functions but expose functions to operate on the class's data. Data structures have no meaningful functions but expose their data instead. Procedural programming is preferred in some cases rather than Object-Oriented programming. It is a trade-off between which programming paradigm to use.

☐ ☐ ☐ ☐ ☐ ☐ ☐

Law of Demeter ([more info](#))

"The Law of Demeter says that a module should not know about the innards of the objects it manipulates." [1]

☐ ☐ ☐ ☐ ☐ ☐ ☐

Error Handling principles

Do you agree with the clean-code [principle]

Fill in how much you agree by using the corresponding Likert scale.

	Strongly Disagree	Disagree	Somewhat Disagree	Neither Agree nor Disagree	Somewhat agree	Agree	Strongly Agree
Prefer Exceptions to Returning Error Codes (more info)	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Throw exceptions rather than returning error codes.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Don't Pass Null (more info)	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Do not pass NULL as an argument to functions.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Don't Return Null (more info)	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Do not return NULL from functions.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Write Your Try-Catch Statement First (more info)	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Beginning with writing the try-catch statement first helps thinking about error handling directly.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

Unit Test principles

Do you agree with the clean-code [principle]

Fill in how much you agree by using the corresponding Likert scale.

Strongly Disagree	Disagree	Somewhat Disagree	Neither Agree nor Disagree	Somewhat agree	Agree	Strongly Agree
-------------------	----------	-------------------	----------------------------	----------------	-------	----------------

Keeping Tests Clean ([more info](#))

Tests need to be kept clean, because they become sort of a mess otherwise.

☐ ☐ ☐ ☐ ☐ ☐ ☐

One Assert per Test ([more info](#))

One assert in each test, not multiple asserts

☐ ☐ ☐ ☐ ☐ ☐ ☐

Single Concept per Test ([more info](#))

One or multiple asserts, but for a single concept is OK

☐ ☐ ☐ ☐ ☐ ☐ ☐

Class principles

Do you agree with the clean-code [principle]

Fill in how much you agree by using the corresponding Likert scale.

Strongly Disagree Disagree Somewhat Disagree Neither Agree nor Disagree Somewhat agree Agree Strongly Agree

Class Organization ([more info](#))

Class organized in the following order:

1. Public static constants
2. Private static variables
3. Private instance/member variables
4. Public functions followed by private functions

☐ ☐ ☐ ☐ ☐ ☐ ☐

High Cohesion ([more info](#))

It is important for modules to achieve high cohesion. High cohesion is concerned with how closely related the connections are within a module or a class.

☐ ☐ ☐ ☐ ☐ ☐ ☐

Low Coupling ([more info](#))

It is important that modules have low coupling. Low coupling refers to a class or module that does not have many connections to other classes or modules.

☐ ☐ ☐ ☐ ☐ ☐ ☐

Encapsulation - Separation of Concerns ([more info](#))

A class should hide some of its behaviour, and data should be kept private (unless it is a data class)

☐ ☐ ☐ ☐ ☐ ☐ ☐

Isolating from Change ([more info](#))

Create interfaces or abstract classes to cope with change.

☐☐☐☐☐☐☐

SRP - Single Responsibility Principle ([more info](#))

"The Single Responsibility Principle (SRP) states that a class or module should have one, and only one, reason to change." [1]

☐☐☐☐☐☐☐

Minimal Classes and Methods ([more info](#))

Do not create too many classes or methods

☐☐☐☐☐☐☐

One Level of Abstraction per Function ([more info](#))

High-abstraction or low-abstraction. Do not intermix these.

☐☐☐☐☐☐☐

Classes should be small ([more info](#))

The responsibilities of a class should be kept low

☐☐☐☐☐☐☐

5 RQ1.a

What are the essential clean code practices and principles you think will help developers write better quality code, and why?

Answer by writing a short text, please

What do you do to write self-explanatory code? or do you need to use comments to explain the code?

Answer with a short text, please

What are the challenges or hindrances with writing self-explanatory code, if any?

Answer with short text, please

I think refactoring is a useful tool to make code clean.

- ☐ Strongly Disagree
- ☐ Disagree
- ☐ Somewhat Disagree
- ☐ Neither Agree nor Disagree
- ☐ Somewhat agree

- ☐ Agree
☐ Strongly Agree

Do you use refactoring as a technique to keep the code clean?

- ☐ No
☐ Yes

How do you use refactoring as a technique to keep the code clean?

Please answer with a short text.

My organization has static analysis tools in place to keep the code clean.

- ☐ No
☐ Yes

Describe your organization's static analysis tool or tools in place to keep the code clean

(Optional)

Please, fill in the name of the tool.

My organization has automatic means not to allow unclean code to go through (e.g., Quality Gate).

- ☐ No
☐ Yes

Which is the automatic tool within the organization used to prevent developers from committing unclean code?

(Optional)

Please, answer the name of the tool with a short text.

Any practice or principle that you see as essential for keeping the code clean, and that has not been mentioned?

(Optional)

Please answer with a short text.

6 RQ2

Do you write clean code initially or write "messy" code that you refactor later?

- ☐ I write clean code initially
- ☐ I write "messy" code and refactor it to make it clean
- ☐ None of these (briefly describe your strategy you use to write clean code)

It is more difficult to write clean code initially

- ☐ Strongly disagree
- ☐ Disagree
- ☐ Somewhat Disagree
- ☐ Neither Agree nor Disagree
- ☐ Somewhat agree
- ☐ Agree
- ☐ Strongly Agree

What do you think are/would be the challenges with writing clean code initially?

Please answer with a short text.

What do you think are/would be the challenges with refactoring unclean code to become clean code?

Please answer with a short text.

What operations or techniques do you usually use to refactor code when needed?

Do you use any IDE/tool that helps with refactoring?

Please, write the name of the IDE/**refactoring tool** used, and a short text describing how it helps you.

I do believe refactoring has a positive effect on the quality of code.

- ☐ Strongly disagree
- ☐ Disagree
- ☐ Somewhat Disagree
- ☐ Neither Agree nor Disagree
- ☐ Somewhat agree
- ☐ Agree
- ☐ Strongly Agree

To write clean code initially, the requirements have to be clearly specified

- ☐ Strongly disagree
- ☐ Disagree
- ☐ Somewhat Disagree
- ☐ Neither Agree nor Disagree
- ☐ Somewhat agree
- ☐ Agree
- ☐ Strongly Agree

It is/would be easier to write clean code at the beginning of a project

- ☐ Strongly disagree
- ☐ Disagree
- ☐ Somewhat Disagree
- ☐ Neither Agree nor Disagree
- ☐ Somewhat agree
- ☐ Agree
- ☐ Strongly Agree

Writing clean code make it easier to make modifications to the code later on

- ☐ Strongly disagree
- ☐ Disagree
- ☐ Somewhat Disagree
- ☐ Neither Agree nor Disagree
- ☐ Somewhat Agree
- ☐ Agree
- ☐ Strongly Agree

I have less time to write clean code towards the end of a project due to deadlines

- ☐ Strongly disagree
- ☐ Disagree
- ☐ Somewhat Disagree
- ☐ Neither Agree nor Disagree
- ☐ Somewhat Agree
- ☐ Agree
- ☐ Strongly Agree

7 RQ2.a

I read and modify source code from other programmers

- ☐ Strongly disagree
- ☐ Disagree
- ☐ Somewhat Disagree
- ☐ Neither Agree nor Disagree
- ☐ Somewhat Agree
- ☐ Agree
- ☐ Strongly Agree

I review or comment on other people's code

- ☐ Strongly disagree
- ☐ Disagree
- ☐ Somewhat Disagree
- ☐ Neither Agree nor Disagree
- ☐ Somewhat Agree
- ☐ Agree
- ☐ Strongly Agree

Other people are reading and modifying the code that I write

- ☐ Strongly disagree
- ☐ Disagree
- ☐ Somewhat Disagree
- ☐ Neither Agree nor Disagree
- ☐ Somewhat Agree
- ☐ Agree
- ☐ Strongly Agree

Other people review or comment the code that I write

- ☐ Strongly disagree
☐ Disagree
☐ Somewhat Disagree
☐ Neither Agree nor Disagree
☐ Somewhat Agree
☐ Agree
☐ Strongly Agree

8 RQ3.a**Rank the code quality characteristics from most important at the top to least important at the bottom**

The quality characteristics have been extracted from [1] and are described as follows:

Readability - readable, no useless code, brevity/conciseness, formatting/layout, style, indentation, the naming convention

Structure - well structured, modular, cohesion, low coupling, no duplication, decomposition

Comprehensibility - understandable, clear purpose

Maintainability - maintainable, adaptable, reusable, used by others, interoperable, portable

Correctness - runnable/free of bugs, language choice, functionally correct (meeting business requirements)

Documentation - documented, commented

Testability - testable, test coverage, automated tests

Dynamic Behavior - robust, good performance, secure

Miscellaneous - license, suitable data structure, metrics/measurements

This is a replication of a question in [1], and therefore we use the same formulation and alternatives.

[1] J. Börstler, H. Störrle, D. Toll, J. Assema, R. Duran, S. Hooshangi, J. Jeuring, H. Keuning, C. Kleiner and B. MacKellar, "I know it when I see it" – Perceptions of Code Quality," in *ITICSE-WGR '17: Proceedings of the 2017 ITICSE Conference on Working Group Reports*, Bologna, Italy, 2017.

1 2 3 4 5 6 7 8 9

Readability ☐ ☐ ☐ ☐ ☐ ☐ ☐ ☐ ☐

Structure	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Comprehensibility	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Maintainability	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Correctness	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Documentation	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Testability	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Dynamic Behavior	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Miscellaneous	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

I do believe that clean code eases the process of reading code.

- ☐ Strongly Disagree
- ☐ Disagree
- ☐ Somewhat Disagree
- ☐ Neither Agree nor Disagree
- ☐ Somewhat agree
- ☐ Agree
- ☐ Strongly Agree

I do believe clean code eases the process of understanding code.

- ☐ Strongly disagree
- ☐ Disagree
- ☐ Somewhat Disagree
- ☐ Neither Agree nor Disagree
- ☐ Somewhat agree
- ☐ Agree
- ☐ Strongly Agree

I do believe that clean code eases the process of reusing code.

- ☐ Strongly disagree
- ☐ Disagree
- ☐ Somewhat Disagree
- ☐ Neither Agree nor Disagree
- ☐ Somewhat agree
- ☐ Agree
- ☐ Strongly Agree

I do believe that clean code eases the process of maintaining code.

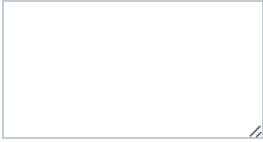
- ☐ Strongly disagree
- ☐ Disagree
- ☐ Somewhat Disagree
- ☐ Neither Agree nor Disagree
- ☐ Somewhat agree
- ☐ Agree
- ☐ Strongly Agree

Writing readable and understandable code is wasting time, and prevents you from being productive and completing tasks.

- ☐ Strongly disagree
- ☐ Disagree
- ☐ Somewhat Disagree
- ☐ Neither Agree nor Disagree
- ☐ Somewhat Agree
- ☐ Agree
- ☐ Strongly Agree

9 RQ3.b

How do you check that your code is readable and understandable by others?

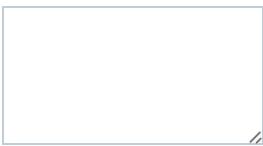


Do you believe that clean code helps with the readability and understandability of code?

- ☐ Strongly Disagree
- ☐ Disagree
- ☐ Somewhat Disagree
- ☐ Neither Agree nor Disagree
- ☐ Somewhat Agree
- ☐ Agree
- ☐ Strongly Agree

Why or why not do you believe clean code helps with readability and understandability?

Please, answer with a short text

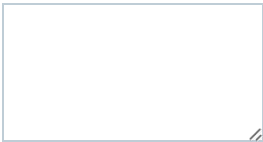


Do you believe that clean code helps with the reusability and maintainability of code?

- ☐ Strongly Disagree
- ☐ Disagree
- ☐ Somewhat Disagree
- ☐ Neither Agree nor Disagree
- ☐ Somewhat Agree
- ☐ Agree
- ☐ Strongly Agree

Why or why not do you believe clean code helps with reusability and maintainability?

Please, answer with a short text



Reading and understanding clean code takes shorter time than reading and understanding the same dirty code.

- ☐ Strongly Disagree
- ☐ Disagree
- ☐ Somewhat Disagree
- ☐ Neither Agree nor Disagree
- ☐ Somewhat Agree
- ☐ Agree
- ☐ Strongly Agree

Reusing clean code takes shorter time than reusing the same dirty code.

- ☐ Strongly Disagree
- ☐ Disagree
- ☐ Somewhat Disagree
- ☐ Neither Agree nor Disagree
- ☐ Somewhat Agree
- ☐ Agree
- ☐ Strongly Agree

Modifying clean code takes shorter time than modifying the same dirty code.

- ☐ Strongly Disagree
 - ☐ Disagree
 - ☐ Somewhat Disagree
 - ☐ Neither Agree nor Disagree
 - ☐ Somewhat Agree
 - ☐ Agree
 - ☐ Strongly Agree
-

10 Endseite

Thank you so much for participating in this survey about Clean Code!

Kevin and Javier

If you have any comment or question, don't hesitate to contact us:

kevinjung98@gmail.com

jgh@bth.se
