Q0.

Survey on Static Analysis Tools: Evaluating the Effectiveness of Code Review Historical Data for Rule Recommendations

We appreciate your participation in this survey, as we aim to assess the effectiveness of utilizing code review historical data for recommending rules in static analysis tools. Your valuable insights and feedback will contribute to enhancing the quality and efficiency of static analysis processes. Please take a few moments to share your experiences and opinions regarding this topic.

We would like to assure you that all the data collected through this survey will be used solely for academic research purposes. Your responses will be treated with the utmost confidentiality and will only be analyzed in an aggregated and anonymized form.

If you have any concerns or questions regarding the data collection process, please feel free to contact us. Thank you for your cooperation and valuable contribution to our academic research.

Q1. How many years of experience do you have in software development?

- ○ Less than 1 year
- ◉ 1-3 years
- ○ 3-5 years
- ○ 5-10 years
- ○ More than 10 years

Q22. Which programming language(s) do you primarily use in your work or development projects? (Select all that apply)

- [x] Java
- [ ] Python
- [x] C/C++
- [ ] Go
- [ ] Ruby
- [ ] JavaScript
- [ ] Others (please specify) _____

Q19. How familiar are you with the practice of code review?

- ○ Very familiar, I regularly participate in code reviews
- ◉ Familiar, I have some experience with code reviews
- ○ Somewhat familiar, I have limited experience with code reviews
- ○ Not familiar, I have little to no experience with code reviews

Q2. Have you ever used static analysis tools in your project to check code quality and identify potential issues?

- (●) Yes
- ( ) No

Q3. If you have used static analysis tools, please choose or list the names or platforms of the tools you have used.

- [ ] SonarQube
- [ ] ESLint
- [ ] PMD
- [x] Checkstyle
- [ ] FindBugs
- [ ] Others

Q3. How frequently do you use static analysis tools in your daily development work?

- (●) Every day
- ( ) Every week
- ( ) Every month
- ( ) Others

Q4. How effective do you think static analysis tools are in improving code quality and identifying potential issues?

- (●) Not effective at all
- ( ) Slightly effective
- ( ) Moderately effective
- ( ) Very effective
- ( ) Extremely effective

Q5. When using static analysis tools, which aspect do you prioritize the most?

- (●) Identifying potential errors or vulnerabilities
- ( ) Checking code conventions and style
- ( ) Performance optimization and resource management
- ( ) Others (Please specify)

Q6. Have you encountered any of the following issues when using static analysis tools? (Select multiple options if applicable)

- [x] Too many false positives (incorrect warnings)
- [x] Hard to understand reports or results
- [ ] Significant performance impact
- [ ] Integration and deployment difficulties
- [ ] Other (please specify)
- [ ] Haven't encountered any issues

Q7. Do you write or refactor code based on the rule set provided by static analysis tools?

- (●) Yes
- ( ) No

Q9. Have you customized the rule set of static analysis tools to meet specific project or team requirements?

- ( ) Yes, I frequently customize the rule set
- (●) Yes, I occasionally customize the rule set
- ( ) No, I have never customized the rule set

Q10. In your opinion, what is the most challenging aspect of customizing the rule set for static analysis tools?

- ( ) Complexity of rule writing and configuration
- (●) Determining the scope of rule set applicable to the project
- ( ) Addressing false positives and false negatives
- ( ) Understanding the impact of rules on code quality
- ( ) Resource and time constraints
- ( ) Other (please specify) [ ]

Q11. Do you think utilizing code review historical data for recommending static rules is feasible? Why?

- [x] Yes, historical data can provide insights into common code issues and patterns
- [ ] Yes, it can help identify recurring issues and improve rule recommendations
- [ ] No, historical data may not accurately represent current code quality
- [ ] Others [ ]

Q12. Do you think utilizing code review for recommending static rules would be helpful for the usage of static analysis tools?

- (●) Yes, it can provide more accurate and targeted static analysis results
- ( ) No, it doesn't have apparent benefits in practical usage
- ( ) Others (Please explain why) [ ]

Q20. When participating in code reviews, which role do you primarily take on in doris project?

- ( ) Developer (who commit the changes in reviews)
- ( ) Reviewer
- (●) Both

Q14. Among the following 30 rules (From SonarQube), half of them consist of the top 15 rules that were selected based on reviewer comments on **apache/doris(https://github.com/apache/doris)** repository that received the most attention. The other half consists of randomly selected rules from the remaining set. Now, please rate the importance of the following rules on a scale of 1 to 5. (The order of the rules presented below is random.)

Deprecated code should be removed		
Method names should comply with a naming convention		
URIs should not be hardcoded		5
Nested blocks of code should not be left empty		5
Two branches in a conditional structure should not have exactly the same implementation		5
Instance methods should not write to "static" fields		5
Boolean literals should not be redundant		5
Using slow regular expressions is security-sensitive		5
Empty statements should be removed		5
Switch cases should end with an unconditional "break" statement		5
"java.nio.Files#delete" should be preferred		
Lambdas should be replaced with method references		5
"toString()" should never be called on a String object		
Generic wildcard types should not be used in return types		5
Overriding methods should do more than simply call the same method in the super class		
Persistent entities should not be used as arguments of "@RequestMapping" methods		
String function use should be optimized for single characters		5
"Object.wait(...)" should never be called on objects that implement "java.util.concurrent.locks.Condition"		5
Deserializing objects from an untrusted source is security-sensitive		
Comma-separated labels should be used in Switch with colon case		5
"@Controller" classes that use "@SessionAttributes" must call "setComplete" on their "SessionStatus" objects		5
Unused assignments should be removed		
Using unsafe Jackson deserialization configuration is security-sensitive		5
Standard outputs should not be used directly to log anything		5
Tests should use fixed data instead of randomized data		
Try-catch blocks should not be nested		5

"readObject" should not be "synchronized"	☆☆☆☆☆	
Alternatives in regular expressions should be grouped when used with anchors	☆☆☆☆☆	
Anonymous classes should not have too many lines	☆☆☆☆☆	5
Short-circuit logic should be used to prevent null pointer dereferences in conditionals	☆☆☆☆☆	5

Q16.

Below are the 15 rules selected based on reviewer comments for doris project.

1. Overriding methods should do more than simply call the same method in the super class
2. Generic wildcard types should not be used in return types
3. "toString()" should never be called on a String object
4. Lambdas should be replaced with method references
5. "java.nio.Files#delete" should be preferred
6. Switch cases should end with an unconditional "break" statement
7. Empty statements should be removed
8. Using slow regular expressions is security-sensitive
9. Boolean literals should not be redundant
10. Instance methods should not write to "static" fields
11. Two branches in a conditional structure should not have exactly the same implementation
12.Nested blocks of code should not be left empty
13. URIs should not be hardcoded
14. Method names should comply with a naming convention
15. Deprecated code should be removed

Do the results align with your expectations (Do the majority of them happen to be the ones that concern you the most?)?

- (●) Yes
- ( ) No (please explain why)

Q13. If you have any additional comments or suggestions, please provide them below. (If it's convenient for you, feel free to provide your contact details.)

Q21. Are you a developer or a user of doris project?

- (●) Developer
- ( ) User
- ( ) Both
- ( ) Not involved

Location Data

Location: (45.8234, -119.7257)

Source: GeoIP Estimation