

/Let a bot deal with your static analysis warnings backlog

EclipseCon 2022
27/10/2022





/TABLE OF CONTENTS



/01 /STATIC ANALYSIS
WARNINGS

/02 /SORALD

/03 /DEMO

/04 /EXPERIMENTS



/WHO ARE WE?



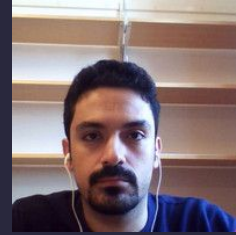
Research Engineer @ KTH



amansha@kth.se



KTH, Stockholm, Sweden



PHD Student @ KTH



khaes@kth.se



KTH, Stockholm, Sweden

/STATIC ANALYSIS WARNING?

*“A potential bug in the program found
without executing it”*

/STATIC ANALYSIS WARNING: EXAMPLE



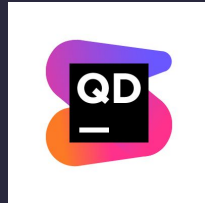
```
String a = null;  
return a.equals("foo");
```

```
String a = null;  
return "foo".equals(a);
```



CALLING `.equals` ON `a` CAN RESULT IN
NULL POINTER EXCEPTION

/HOW TO FIND THEM?

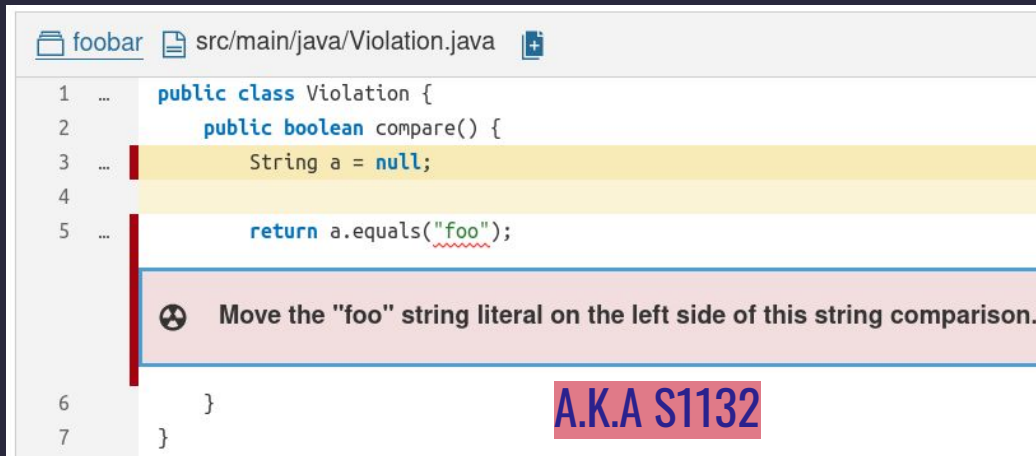


!! Method invocation 'equals' will produce 'NullPointerException'

src/main/java/Violation.java

```
3      String a = null;  
4  
5      return a.equals("foo");  
6  }  
7  }
```

/HOW TO FIND THEM?



The screenshot shows a code editor window for a file named `src/main/java/Violation.java`. The code contains a `public class Violation` with a `compare()` method. Inside the method, there is a line `String a = null;` and a `return a.equals("foo");` statement. A red squiggly line is under the `"foo"` string literal. Below the code, a red tooltip message with a warning icon says: "Move the 'foo' string literal on the left side of this string comparison." To the right of the tooltip, the text "A.K.A S1132" is displayed in a red box.

```
1 ... public class Violation {  
2     public boolean compare() {  
3         String a = null;  
4  
5         return a.equals("foo");  
6     }  
7 }
```

A.K.A S1132

< IT IS ALSO A
!BUG! >



“LOTS OF OTHER THINGS TO DO RIGHT NOW”



> Something more actionable is required.



SpoonLabs/sorald



Automatic repair system for static analysis warnings
from SonarQube's SonarJava

18

Contributors



4

Used by



60

Stars



18

Forks



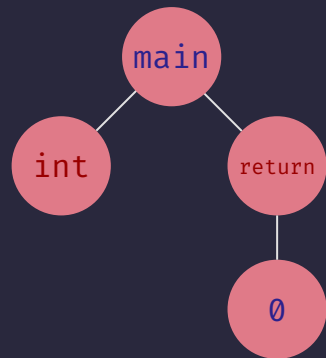


It ain't much, but it's honest work

/ABSTRACT SYNTAX TREE

```
int main() {  
    return 0;  
}
```

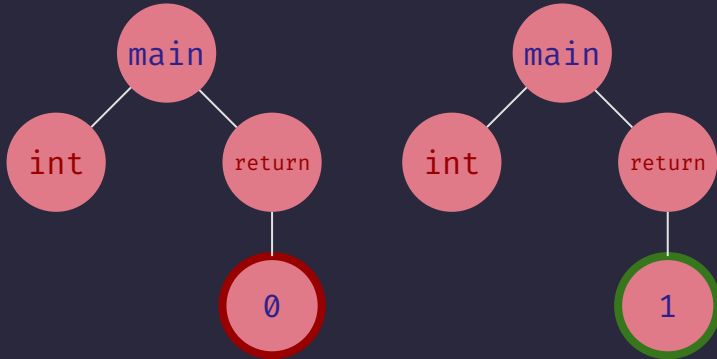
Abstract Syntax Tree (AST): tree
to represent the structure of
the program



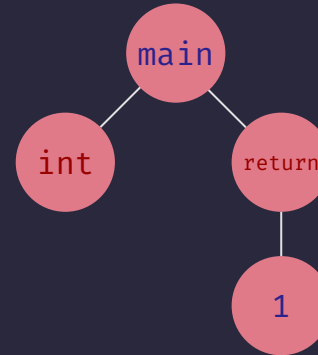
/SPOON



/AST TRANSFORMATION



/HIGH FIDELITY PRETTY-PRINTING



```
int main() {  
    return 1;  
}
```



/HOW SORALD WORKS: DEEP DIVE

Hey, Sorald! Repair this for me, please?

04:20

Sorald

On it. @sonarqube, do you find violations?

04:21

SonarQube

Yes. I find S1132 at line 3 column 8.

04:21

Sorald

Let me do the AST transformation now :)

04:21

Sorald

@spoon, please make it pretty. :*

04:21

Spoon

Sure thing. @user, here you go!

04:22

So grateful to all of you <3

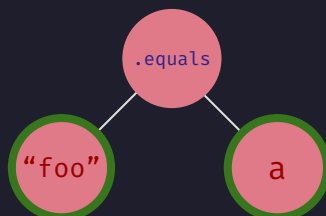
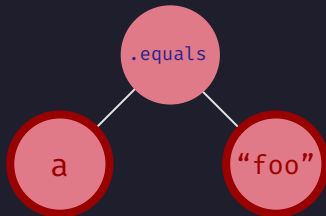
04:22



/DEEP DIVE CONCLUSION



```
1 String a = null;  
2  
3 return a.equals("foo");
```



```
1 String a = null;  
2  
3 return "foo".equals(a);
```

/SOURCE CODE EXAMPLE

```
·@Override  
·protected void repairInternal(CtInvocation<?> element)·{  
·····//·Get·executable·reference·of·method·to·be·called  
·····CtExecutableReference<?> ctExecutableReferenceToMethodToBeCalled = element.getExecutable();  
  
·····//·Create·a·new·invocation·in·which·the·receiver·and·argument·are·swapped  
·····CtInvocation<?> newInvocation =  
···········getFactory()  
············Code()  
············createInvocation(  
·······················element.getArguments().get(0),  
·······················ctExecutableReferenceToMethodToBeCalled,  
·······················element.getTarget());  
  
·····//·Replace·the·old·invocation·by·the·new·one  
·····element.replace(newInvocation);  
  
·····//·Delete·the·null·check·on·the·variable·if·it·exists  
·····deleteNullCheckIfExists(newInvocation);
```


/WAYS TO USE SORALD



```
$ java -jar sorald.jar repair
```



```
$ mvn sorald:repair
```



fix: Fix violation of Sonar rule 2097 #3972

Merged by monperrus INRIA:master ← slarse:fix-2097-violation on Jun 1, 2021

Choose your favourite:



/DEMO (STEPS)



/Mining

We use Sorald's miner to find violations



/Repairing

We use Sorald's repair to fix violations



/Re-mining

We mine violations again to check if they are fixed



/Compile

We compile to ensure we have not introduced an error

/DEMO (COMMANDS)

Let's see how Sorald works in action:

- We repair a project by SonarSource: `Sonar-Scanner-Cli@5c518d6`
<https://github.com/SonarSource/sonar-scanner-cli>

- We use Sorald's maven plugin

- We fix violations of S1132

```
$ mvn se.kth.castor:sorald:repair -DruleKey=S1132
```

- We re-mine violations and compile the project after the repair

- S1132: "Strings literals should be placed on the left side when checking for equality"

- Solution: Move the literal to the left

/EXPERIMENTING IN THE WILD



/161 Popular Repos

We select 161 Github repos:

- More than 50 stars
- PR & CI friendly
- Active & Maven
- Healthy (mvn compile)



/10 Sonar Rules

We experiment with 10 rules:

- Bug type
- Common in past experiments

/EXPERIMENTING IN THE WILD: SELECTED RULES

1. `Thread.run()` should not be called directly.
2. Synchronization should not be based on Strings or boxed primitives.
3. Resources should be closed.
4. `BigDecimal(double)` should not be used.
5. `hashCode` and `toString` should not be called on array instances.
6. `InterruptedException` should not be ignored.
7. Math operands should be cast before assignment.
8. `toString()` and `clone()` methods should not return null.
9. `Iterator.next()` methods should throw `NoSuchElementException`.
10. Strings and Boxed types should be compared using `equals()`.

<65%>

> Violations fixed in 161 projects <



54 patches for 80 newly
introduced violations in 350
days

Courtesy: eclipse/repairnator

17

Number of merged PRs

10

Number of closed PRs

2

Number of pending PRs



stevehu commented on Dec 4, 2021

Contributor



@khaes-kth This is a very good tool and it would help a lot of open-source projects. The PR looks pretty good and easy to understand with a detailed description. I like the reference links if users want to explore the extra details.

Regarding the coding style, I found the first file has some indentation issues with some extra spaces. Other than that, looks good.

Again, thanks a lot for your help.



1



A stylized illustration of a computer monitor with a white screen and a pinkish-orange base. On the screen, the word 'sonarlint' is written in a bold, black, sans-serif font. A red wavy line is positioned under the 'i' in 'sonarlint'.

sonarlint

/RELATED TOOL

<https://www.sonarsource.com/products/sonarlint/>

/FUTURE IMPROVEMENTS



/Add More Analyzers

There are other analyzers to be plugged in:

- Qodana
- SpotBugs
- ...



/Filter Out Unacceptable Patches

Unacceptable patches can be filtered out:

- False positive warnings
Eg. `str1==str2`
- Uncertain fixes

<AUTOMATIC REPAIR TOOL>

<AST TRANSFORMATIONS>

<PART OF DEVELOPMENT PROCESS>

/THANKS!

amansha@kth.se

khaes@kth.se

Link to GitHub repository:

<https://github.com/SpoonLabs/sorald>

White Paper:

<https://ieeexplore.ieee.org/document/9756950>

Voting Link



CREDITS: This presentation template was created by Slidesgo, and includes icon by Flaticon, and infographics & images by Freepik