Nullness_Lite

Anny Kong (yk57)
Spencer Huang (yh73)
Alice Zhao (zhaox29)
Cassie Chen (chenm32)

Motivation

- NullPointerException: the most frequent bug in Java Program [1]
- Nullness Bug Detector: a huge convenience for developers

Implementation and Evaluation

- Our Lite Version of Nullness Checker: **Nullness_Lite**

- Nullness Checker of Checker Framework
- NullAway
- FindBugs
- Built-ins of Eclipse and IntelliJ

Related works

NullAway

- Fast: build time overhead of running NullAway < 10% [2]
- Easy to Use: plugin of ErrorProne which is widely used in industry
- Limitations: does not check code using generics and null assertions [3 & 4]

IntelliJ and Eclipse

- Statically checks errors when users type
- Provide Their Own Libraries of Annotations
- “Infer Nullity”: IntelliJ automatically introduces @Nullable and @NotNull to project

Related works

FindBugs

- Uses heuristic pattern-matching to analyze
- Powerful: directly analyzes bytecode (does not need source code)

Nullness Checker

- A **sound**, pluggable type checker which aims to find all nullness bugs (type-based dataflow analysis) [5]
- Ground Truth of our evaluation
- Conservative: more false positives found [6]

Our Expectation of Nullness_Lite

1. Allow users to perform fast verification before full analysis (but the result would be unsound in this case)
2. Reduce the number of false positives.

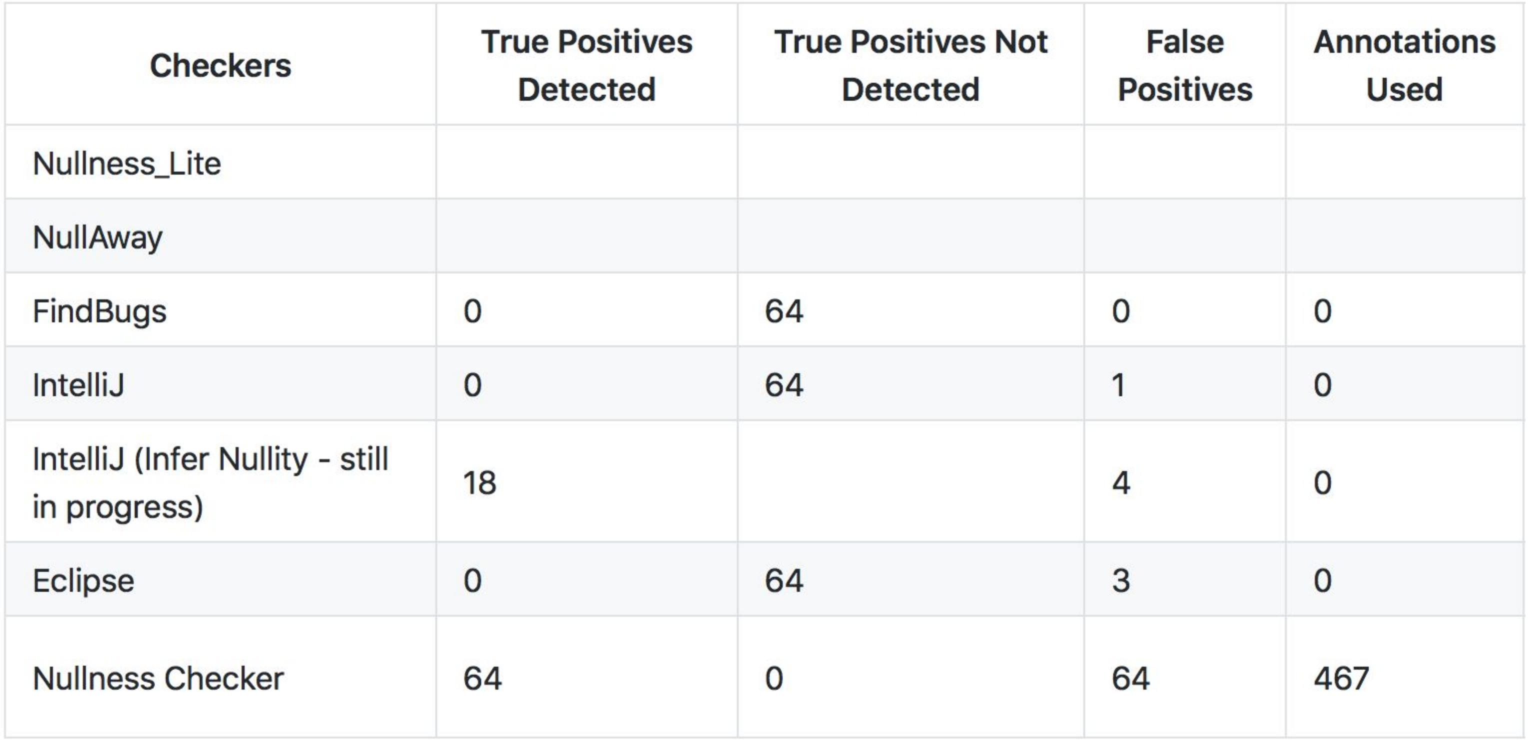
Implementation of Nullness_lite

1. Disable part of the features of the Initialization Checker (assume all values are initialized)
2. Disable part of the features of the Map Key Checker (assume all keys exist in the map, which means get(key) will never return null)
3. Modify some assumptions of the dataflow analysis (assume all methods are SideEffectFree & no aliasing)
4. Modify the behaviors of boxing primitives (assume boxing of primitives, valueOf() is @Pure)

Evaluation

- Program for Evaluation: JUnit4
- Number of Annotations Added: (1) errors found by checkers (2) we add annotations in order to eliminate some of the errors
- False Positives and True Positives:
  (1) manually reasoning
  (2) less false positives: more flexible and user-friendly
- Bugs Not Revealed: Nullness Checker is the "ground truth"

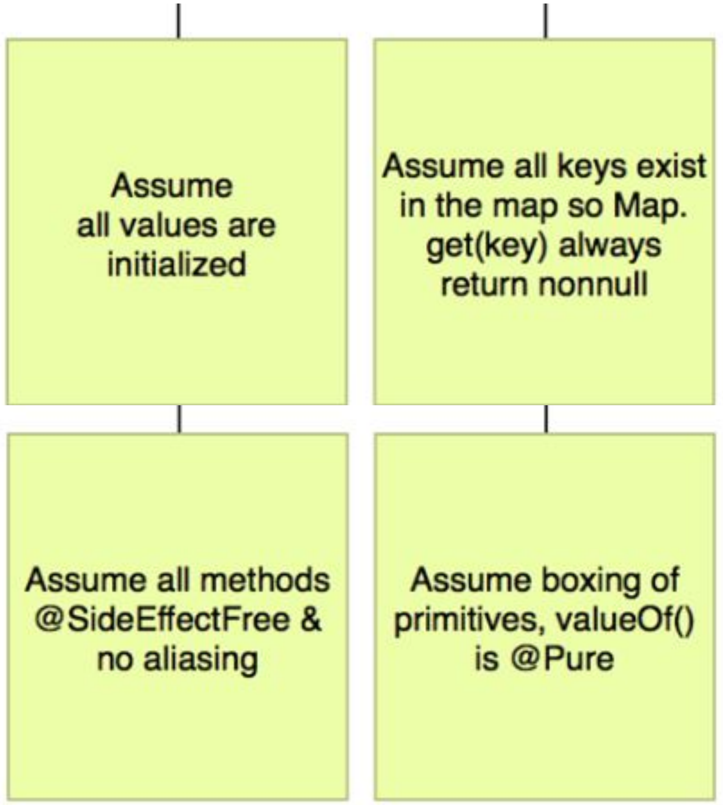
Evaluation



Checkers	True Positives Detected	True Positives Not Detected	False Positives	Annotations Used
Nullness_Lite				
NullAway				
FindBugs	0	64	0	0
IntelliJ	0	64	1	0
IntelliJ (Infer Nullity - still in progress)	18		4	0
Eclipse	0	64	3	0
Nullness Checker	64	0	64	467

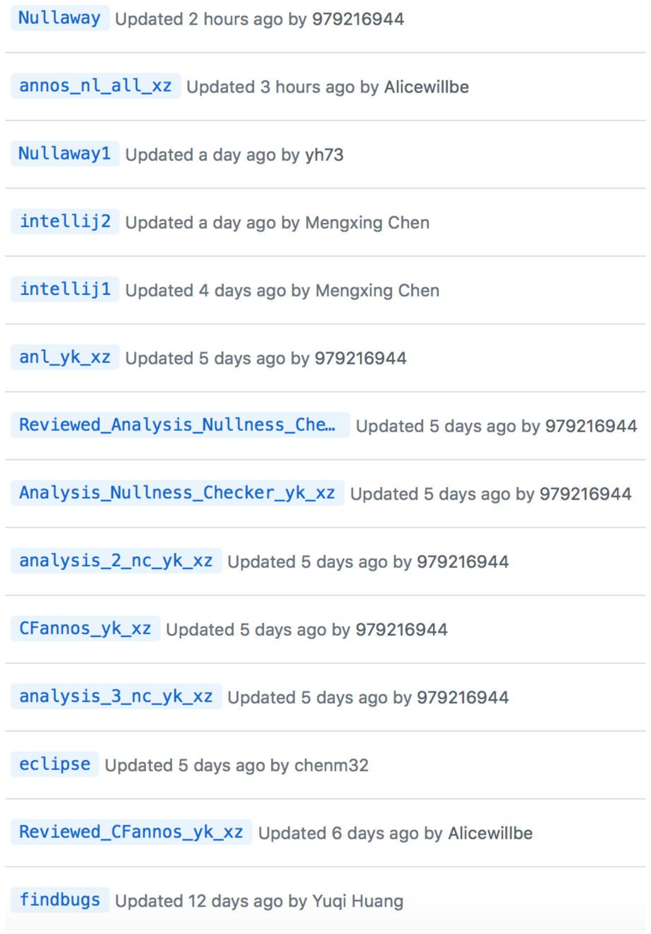
Evaluation

- Which Feature to Disable?
  - We choose 4 features of Nullness Checker
  - Disable a feature if:
    - Fewer annotations added
    - Fewer false positives reported


Assume all values are initialized
Assume all keys exist in the map so Map. get(key) always return nonnull
Assume all methods @SideEffectFree & no aliasing
Assume boxing of primitives, valueOf() is @Pure

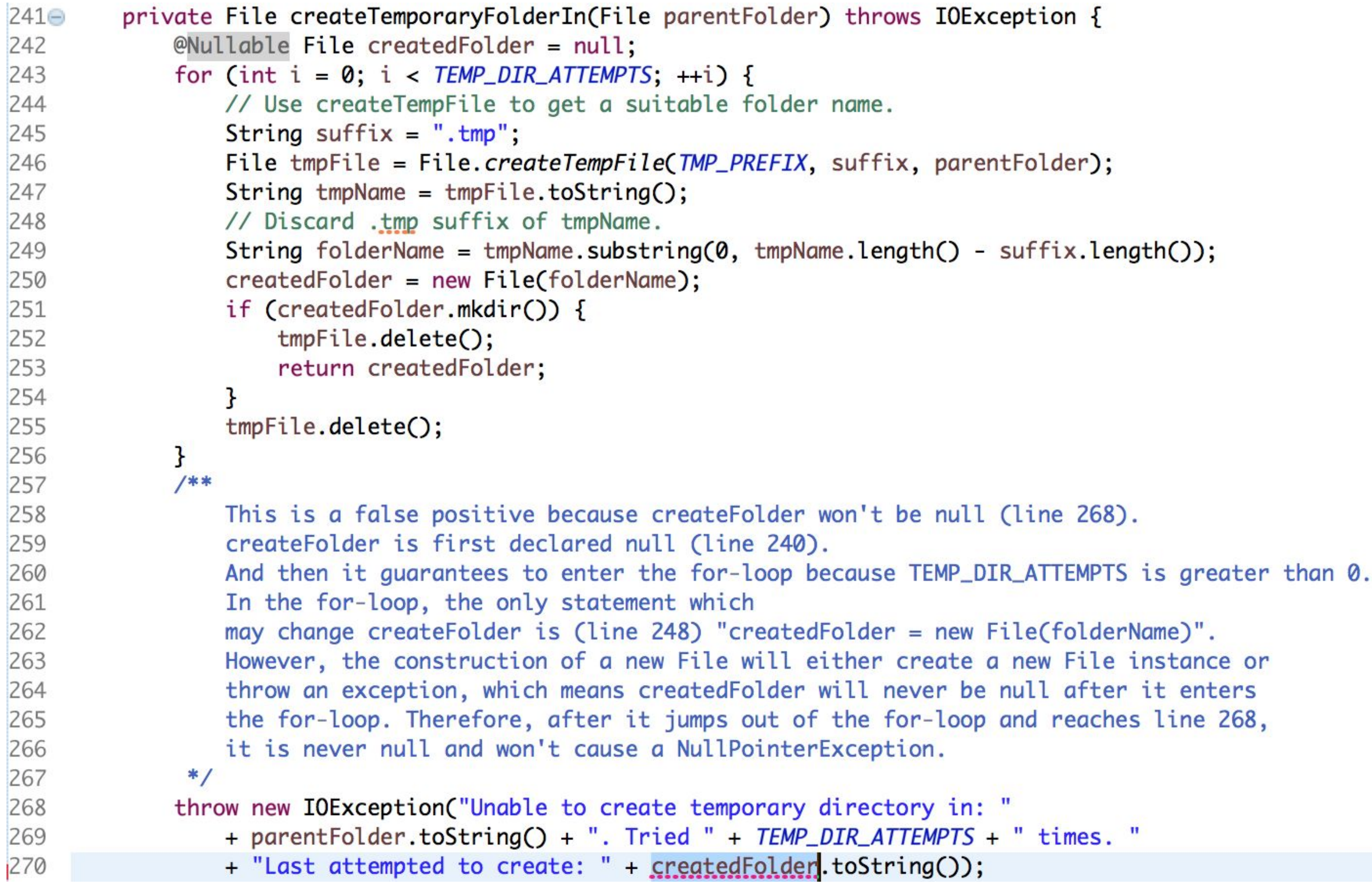
Preliminary result

- Our fork of JUnit4: https://github.com/NullnessLiteGroup/junit4
- One branch for each checker that we need to evaluate
  - Two for IntelliJ and four for Nullness_Lite
- Script files (or manuals) for reproduction



Nullaway Updated 2 hours ago by 979216944
annos_nl_all_xz Updated 3 hours ago by Alicewillbe
Nullaway1 Updated a day ago by yh73
intellij2 Updated a day ago by Mengxing Chen
intellij1 Updated 4 days ago by Mengxing Chen
anl_yk_xz Updated 5 days ago by 979216944
Reviewed_Analysis_Nullness_Che… Updated 5 days ago by 979216944
Analysis_Nullness_Checker_yk_xz Updated 5 days ago by 979216944
analysis_2_nc_yk_xz Updated 5 days ago by 979216944
CFannos_yk_xz Updated 5 days ago by 979216944
analysis_3_nc_yk_xz Updated 5 days ago by 979216944
eclipse Updated 5 days ago by chenm32
Reviewed_CFannos_yk_xz Updated 6 days ago by Alicewillbe
findbugs Updated 12 days ago by Yuqi Huang

EXAMPLE



```
241     private File createTemporaryFolderIn(File parentFolder) throws IOException {
242         @Nullable File createdFolder = null;
243         for (int i = 0; i < TEMP_DIR_ATTEMPTS; ++i) {
244             // Use createTempFile to get a suitable folder name.
245             String suffix = ".tmp";
246             File tmpFile = File.createTempFile(TMP_PREFIX, suffix, parentFolder);
247             String tmpName = tmpFile.toString();
248             // Discard .tmp suffix of tmpName.
249             String folderName = tmpName.substring(0, tmpName.length() - suffix.length());
250             createdFolder = new File(folderName);
251             if (createdFolder.mkdir()) {
252                 tmpFile.delete();
253                 return createdFolder;
254             }
255             tmpFile.delete();
256         }
257         /**
258             This is a false positive because createFolder won't be null (line 268).
259             createFolder is first declared null (line 240).
260             And then it guarantees to enter the for-loop because TEMP_DIR_ATTEMPTS is greater than 0.
261             In the for-loop, the only statement which
262             may change createFolder is (line 248) "createdFolder = new File(folderName)".
263             However, the construction of a new File will either create a new File instance or
264             throw an exception, which means createdFolder will never be null after it enters
265             the for-loop. Therefore, after it jumps out of the for-loop and reaches line 268,
266             it is never null and won't cause a NullPointerException.
267          */
268         throw new IOException("Unable to create temporary directory in: "
269             + parentFolder.toString() + ". Tried " + TEMP_DIR_ATTEMPTS + " times. "
270             + "Last attempted to create: " + createdFolder.toString());
```

Thank you

Works cited

1. Maciej Cielecki, Je ̨drzej Fulara, Krzysztof Jakubczyk, and Jancewicz. Propagation of JML non-null annotations in java programs. In *Principles and practice of programming in Java*, pages 135–140, 2006.

2. Uber. "Uber/NullAway." *GitHub*, github.com/uber/NullAway.

3. Sridharan, Manu. "Support Models of Generic Types · Issue #54." *GitHub, Uber/NullAway*, 6 Nov. 2017, github.com/uber/NullAway/issues/54.

4. kevinzetterstrom. "Support for null assertions · Issue #122 · Uber/NullAway." GitHub, github.com/uber/NullAway/issues/122.

5. Dietl, Werner, et al. "Building and Using Pluggable Type-Checkers." Proceeding of the 33rd International Conference on Software Engineering - ICSE '11, 2011, doi:10.1145/1985793.1985889

6. Dietl, Werner, and Michael Ernst. " Preventing Errors Before They Happen The Checker Framework." *Preventing Errors Before They Happen The Checker Framework*, www.google.com/url?sa=t&rct=j&q=&esrc=s&source=web&cd=7&cad=rja&uact=8&ved=0ahUKEwiup6_V5bPaAhXKrFQKHW-gCxYQFghaMAY&url=https%3A%2F%2Fstatic.rainfocus.com%2Foracle%2Foow17%2Fsess%2F1492901668615001brln%2FPF%2F2017-10-02%2520CF%2520%40%2520JavaOne_1507012791774001WJ2t.pdf&usg=AOvVaw3mAtzExTzYm6gr3sCn0cXb.