Usage and Attribution of

*stack*overflow* Code Snippets in *GitHub* Projects

Sebastian Baltes
@s_baltes

copyright-softw.engineering

41st International Conference on Software Engineering (ICSE 2019)
May 29-31, Montreal, Canada
Thanks to my co-author!

Usage and attribution of Stack Overflow code snippets in GitHub projects

Sebastian Baltes¹ ID - Stephan Diehl¹ ID

Published online: 01 October 2018
© Springer Science+Business Media, LLC, part of Springer Nature 2018

Abstract
Stack Overflow (SO) is the most popular question-and-answer website for software developers, providing a large amount of copyable code snippets. Using those snippets raises maintenance and legal issues. SO’s license (CC BY-SA 3.0) requires attribution, i.e., referencing the original question or answer, and requires derived work to adopt a compatible license. While there is a heated debate on SO’s license model for code snippets and the
"You must give **appropriate credit** [...] and indicate if changes were made."

“If you [...] **build upon** the material, you must **distribute your contributions** under the same license as the original.”
Results from our Online Surveys

• **46%** of the participants admitted copying code from Stack Overflow **without attribution**

• **75%** did **not know** that content on SO is licensed under **CC BY-SA**

• **67%** did **not know** that **attribution is required**

→ Lack of awareness
Background

“Well, but these snippets are rather trivial and not protected by copyright.”

• Not all code snippets on Stack Overflow are copyrightable.

• “A snippet that is more than one or two lines of standard function calls would typically be creative enough for copyright” [Engelfriet 2016]

• But no “international standard for originality” [Creative Commons 2017b]
Here's what I do:

1. First of all check what providers are enabled. Some may be disabled on the device, some may be disabled in application manifest.
2. If any provider is available I start location listeners and timeout timer. It's 20 seconds in my example, may not be enough for GPS so you can enlarge it.
3. If you get update from location listener I use the provided value. I stop listeners and timer.
4. If I don't get any updates and timer elapses I have to use last known values.
5. I grab last known values from available providers and choose the most recent of them.

Here's how I use my class:

```java
import java.util.Timer;
import java.util.TimerTask;
import android.content.Context;
import android.location.Location;
import android.location.LocationManager;

public class MyLocation {
    Timer timer;
    LocationManager lm;
    LocationResult result;

    public boolean getlocation(Context context, LocationResult locationResult) {
        // use locationlistener callback class to pass location value from MyLocation to some code.
        if (lm == null) {
            lm = (LocationManager) context.getSystemService(Context.LOCATION_SERVICE);
        }
        // Network will be thrown if provider is not permitted
        try {
            lm.requestLocationUpdates(0, 5000, 10, new LocationListener() {
                public void onLocationChanged(Location location) {
                    // use Location object
                    result.setLocation(location);
                    // send Location object to some code.
                }
            });
        } catch (SecurityException | IllegalArgumentException e) {
            // handle exception
        }
    }
}
```

Here's my Location class:

```java
public class LocationResult {
    public Location location; // use Location object
    public NetworkResult networkResult; // use NetworkResult object
}
```

Somebody may also want to modify my logic. For example if you get update from Network provider don't stop listeners but continue waiting. GPS gives more accurate data so it's worth waiting for it. If timer elapses and you've got update from Network but not from GPS then you can use value provided from Network.

One more approach is to use LocationClient. If developer.android.com/trainglocation notice current.html It requires Google Play Services app to be installed on user device.

https://stackoverflow.com/a/3145655

Stack Overflow Code Snippets in GitHub Projects (ICSE 2019 J1C2)
Stack Overflow Code in the OpenJDK

Get rid of the humanReadableByteCount() method in openjdk/hotspot

Details
Type: Bug
Priority: P2
Affects Version/s: 9
Component/s: hotspot
Status: RESOLVED
Resolution: Fixed
Fix Version/s: 9

implement the method humanReadableByteCount which body was copied from the Stack Overflow site: https://stackoverflow.com/a/3758880

It's just a few lines of code, but it **could cause legal issues**. The method should be either re-implemented or removed.

Besides the potential legal issues, duplicating a code is **not a good practice**.

https://bugs.openjdk.java.net/browse/JDK-8170860
Implications of Stack Overflow’s License

Permissive Licenses

• Permit using the licensed source code in proprietary software without publishing changes or the derived work
• Examples: MIT, Apache, and BSD license families

Copyleft Licenses

• Requires either modifications to the licensed content or the complete derived work to be published under the same or a compatible license (share-alike)
• Examples (weak copyleft): Mozilla/Eclipse Public Licenses
• Examples (viral copyleft): GNU General Public Licenses, Creative Commons Share-Alike Licenses (e.g., CC BY-SA)
Enforceability of Copyleft Licenses

• Courts in the US and Europe ruled that open source licenses are **enforceable contracts**

• Authors are able to **sue** when terms such as the share-alike requirement are violated:
  • **Interdict distribution** of derived work
  • **Claim monetary damages**

• USA: DMCA takedown notices for allegedly infringed copyright
  • Example: [https://github.com/github/dmca](https://github.com/github/dmca)

• Risk in mergers and acquisitions of companies
  • Example: FSF vs. Cisco lawsuit
Research Question

Question: How frequently is code from Stack Overflow posts used in public GitHub projects without the required attribution?

Approach: Triangulate an estimate for the attribution ratio using three different methods.
Method 1: Regular Expressions

GitHub

209m files in 4.1m projects

13m Java files in 336k projects

10 most frequently referenced answers

Check if attributed
(URL to answer or corresponding question)

Check for false positives

4,198 files with matches

Manually build regular expressions matching code snippets
(referenced usages as test cases)

Check external availability

...stackoverflow.com...

209m files in 4.1m projects

13m Java files in 336k projects

10 most frequently referenced answers

Check if attributed
(URL to answer or corresponding question)

Check for false positives

4,198 files with matches

Manually build regular expressions matching code snippets
(referenced usages as test cases)

Check external availability

...stackoverflow.com...

209m files in 4.1m projects

13m Java files in 336k projects

10 most frequently referenced answers

Check if attributed
(URL to answer or corresponding question)

Check for false positives

4,198 files with matches

Manually build regular expressions matching code snippets
(referenced usages as test cases)

Check external availability

...stackoverflow.com...

209m files in 4.1m projects

13m Java files in 336k projects

10 most frequently referenced answers

Check if attributed
(URL to answer or corresponding question)

Check for false positives

4,198 files with matches

Manually build regular expressions matching code snippets
(referenced usages as test cases)

Check external availability

...stackoverflow.com...
## Results

<table>
<thead>
<tr>
<th>Rank</th>
<th>Matches</th>
<th>Recall</th>
<th>Attribution</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>ALL</td>
<td>REF/F_AQ</td>
<td>REF/DISTINCT</td>
</tr>
<tr>
<td>1</td>
<td>997</td>
<td>79.5%</td>
<td>21.7%</td>
</tr>
<tr>
<td>2</td>
<td>1,843</td>
<td>60.0%</td>
<td>6.6%</td>
</tr>
<tr>
<td>3</td>
<td>2,662</td>
<td>80.6%</td>
<td>9.6%</td>
</tr>
<tr>
<td>4</td>
<td>420</td>
<td>94.7%</td>
<td>10.6%</td>
</tr>
<tr>
<td>5</td>
<td>1,492</td>
<td>73.5%</td>
<td>6.2%</td>
</tr>
<tr>
<td>6</td>
<td>2,642</td>
<td>87.8%</td>
<td>8.1%</td>
</tr>
<tr>
<td>7</td>
<td>160</td>
<td>29.3%</td>
<td>9.7%</td>
</tr>
<tr>
<td>8</td>
<td>355</td>
<td>61.1%</td>
<td>12.6%</td>
</tr>
<tr>
<td>9</td>
<td>295</td>
<td>10.6%</td>
<td>2.2%</td>
</tr>
<tr>
<td>10</td>
<td>65</td>
<td>42.3%</td>
<td>33.3%</td>
</tr>
<tr>
<td>All</td>
<td>10,931</td>
<td>M 61.9%</td>
<td>M 12.1%</td>
</tr>
</tbody>
</table>
Method 2: Code Clone Detector

• **Goal:** Use code clone detector to find clones of a sample of Stack Overflow snippets in a sample of GitHub projects

• **Why samples?**
  • Code clone detection is computationally expensive

• **Which snippets and projects to select?**
  • Random samples: Many *toy projects* on GitHub and many *irrelevant snippets* on Stack Overflow
  • Purposive sampling: Limited generalizability
GitHub Project Sample

- Focus on **popular** GitHub projects
- High precision in selecting “engineered” software projects [Munaiah et al. 2017]
- Greater (potential) impact of licensing issues

Sample size: 3,000 / 2,313
Stack Overflow Snippet Samples

• Non-trivial snippets retrieved from 100 most frequently referenced answers (n=111)
  \[ \Rightarrow S_{\text{top}100} \]

• Non-trivial snippets retrieved from answers referenced in GitHub projects (n=137)
  \[ \Rightarrow S_{\text{gh}} \]

• External sources: Only three snippets available under a more permissive license than CC BY-SA
Code Clone Detector Calibration

Comparison of CPD configurations

https://pmd.github.io/
## Results

<table>
<thead>
<tr>
<th>Set</th>
<th>Snippets</th>
<th>Files</th>
<th>Repos</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>ALL</td>
<td>MATCHED</td>
<td>ANSWERS</td>
</tr>
<tr>
<td>$S_{gh}$</td>
<td>137</td>
<td>53 (39%)</td>
<td>102</td>
</tr>
<tr>
<td>$S_{top100}$</td>
<td>111</td>
<td>48 (43%)</td>
<td>85</td>
</tr>
<tr>
<td>$\cup S$</td>
<td>222</td>
<td>101 (46%)</td>
<td>169</td>
</tr>
</tbody>
</table>
Method 3: Exact Matches

• **Goal:** Address shortcomings of Method 1 and 2
  • Increase sample sizes
  • Exclude snippets available on external sources
  • Systematically exclude short snippets

• Select as many projects and snippets as possible and search for (almost) exact matches
Method 3: Exact Matches

GitHub

209m files in 4.1m projects

- Project is not a fork, has ≥ 5 Java files and ≥ 1 watcher(s)
- File has ending .java has ≥ 68 NLOC (Q₃)

21m answers

- Question tagged java or android
- Answer score ≥ 10
- Code block ≥ 6 NLOC

Google Cloud

1.7m Java files in 64k projects

Normalization and substring search

10,358 matches

Stack Overflow

29k snippets from 24k answers

18k209m files in 4.1m projects

- Project is not a fork, has ≥ 5 Java files and ≥ 1 watcher(s)
- File has ending .java has ≥ 68 NLOC (Q₃)

21m answers

- Question tagged java or android
- Answer score ≥ 10
- Code block ≥ 6 NLOC

Google Cloud

1.7m Java files in 64k projects

Normalization and substring search

10,358 matches

Stack Overflow

29k snippets from 24k answers

18
Method 3: Filtering of Matches

- Use heuristic to detect and exclude matches in mirrors of JDK and Android source code.
- Manually analyze answers, exclude snippets that are too trivial, incomplete, or copied from an external source.
- Use GitHub API to remove matches where commit adding snippet is older than answer on Stack Overflow.

10,358 matches

- Only 7.6% attributed.

Check if attributed (URL to answer or corresponding question).

1,379 matches

1,369 matches
Attribution

**Attribution ratio:**
- Method 1 (regular expressions): 23 %
- Method 2 (code clone detector): 24 %
- Method 3 (exact matches): 8 %

**Conservative estimate:**
- **Attribution ratio** $\leq$ 25%
Only **2%** of all analyzed repositories (all methods) containing code from Stack Overflow attributed its source and used a **compatible license** (not CC BY-SA, but GPL 3.0).

<table>
<thead>
<tr>
<th>SPDX license name</th>
<th>Number of repos containing a SO code snippet clone that was unattributed (n = 2,962)</th>
<th>attributed (n = 329)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Apache-2.0</td>
<td>921 (31.1%)</td>
<td>99 (30.1%)</td>
</tr>
<tr>
<td>MIT</td>
<td>621 (21.0%)</td>
<td>72 (21.9%)</td>
</tr>
<tr>
<td>GPL-3.0</td>
<td>430 (14.7%)</td>
<td>66 (18.6%)</td>
</tr>
<tr>
<td>GPL-2.0</td>
<td>284 (9.6%)</td>
<td>21 (6.4%)</td>
</tr>
<tr>
<td>BSD-3-Clause</td>
<td>93 (3.1%)</td>
<td>9 (2.7%)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SPDX license name</th>
<th>Number of repos containing a SO code snippet clone that was unattributed (n = 144)</th>
<th>attributed (n = 55)</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>56 (38.9%)</td>
<td>18 (32.7%)</td>
</tr>
<tr>
<td>Apache-2.0</td>
<td>31 (21.9%)</td>
<td>15 (27.3%)</td>
</tr>
<tr>
<td>GPL-3.0</td>
<td>17 (11.8%)</td>
<td>6 (10.9%)</td>
</tr>
<tr>
<td>MIT</td>
<td>6 (4.2%)</td>
<td>4 (7.3%)</td>
</tr>
<tr>
<td>GPL-2.0</td>
<td>4 (2.8%)</td>
<td>2 (3.6%)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SPDX license name</th>
<th>Number of repos containing a SO code snippet clone that was unattributed (n = 1,169)</th>
<th>attributed (n = 163)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Apache-2.0</td>
<td>353 (30.2%)</td>
<td>36 (27.4%)</td>
</tr>
<tr>
<td>MIT</td>
<td>339 (29.0%)</td>
<td>25 (17.7%)</td>
</tr>
<tr>
<td>GPL-3.0</td>
<td>211 (18.0%)</td>
<td>19 (11.7%)</td>
</tr>
<tr>
<td>None</td>
<td>153 (13.1%)</td>
<td>61 (37.9%)</td>
</tr>
<tr>
<td>GPL-2.0</td>
<td>89 (7.6%)</td>
<td>8 (4.8%)</td>
</tr>
</tbody>
</table>
Reaching out to Developers

• **Contacted owners** of GitHub projects containing copies of Stack Overflow snippets

• **75% not aware** of CC BY-SA licensing  
  (see slide about online surveys)

• Many thankful responses
Usage and Attribution of Stack Overflow Code Snippets in GitHub Projects

Sebastian Baltes
@s_baltes

Data and scripts available on Zenodo

snippets.sbaltes.com

Stack Overflow Code Snippets in GitHub Projects (ICSE 2019 J1C2)