Augmenting Stack Overflow with API Usage Patterns Mined from GitHub

Anastasia Reinhart\(^1,2\)* Tianyi Zhang\(^1\) Mihir Marthur\(^1\) Miryung Kim\(^1\)

\(^1\)University of California, Los Angeles
\(^2\)George Fox University

* Work done as a research intern at UCLA.
Using APIs properly is becoming a key challenge

e.g., JDK APIs

Android SDK

Apache Spark

PyTorch

TensorFlow
The Status Quo of Learning APIs

Developers often search online for code examples to learn APIs [Sadowski et al. 2016]
The Limitation of Online Code Examples

• Programmers can only inspect a handful of search results. [Brandt et al., 2009, Starke et al., 2009, Duala-Ekoko and Robillard, 2012]

• Individual code examples may suffer from
  – insecure coding practices [Fischer et al., 2017]
  – unchecked obsolete usage [Zhou and Walker, 2016]
  – low readability [Treude and Robillard, 2017]
The Limitation of Online Code Examples

- Programmers can only inspect a handful of search results. [Brandt et al., 2009, Starke et al., 2009, Duala-Ekoko and Robillard, 2012]

A recent study shows that 31% of SO posts have potential API usage violations. [Zhang et al., Are Online Code Examples Reliable? A Study of API Misuse on Stack Overflow, ICSE 2018]

Dataset: http://web.cs.ucla.edu/~tianyi.zhang/examplecheck.html
Missing If Checks

I think you have declared

```java
Map map = new TreeMap();
```

you need to instead do

```java
TreeMap map = new TreeMap(); // OR SortedMap map = new TreeMap()
map.firstKey();
```

Because `firstKey()` is a method which exists in `TreeMap` which is not defined in `Map` contract (or interface)

https://stackoverflow.com/questions/21983867
Missing If Checks

I think you have declared

```java
Map map = new TreeMap();
```
you need to instead do

```java
TreeMap map = new TreeMap(); // OR SortedMap map = new TreeMap()
map.firstKey();
```

Because `firstKey()` is a method which exists in `TreeMap` which is not defined in `Map` contract (or interface)

This example throws `NoSuchElementException`. You should not call `firstKey` on an empty `TreeMap`.
Missing API Calls

You might want to take a look at `BitSet`.

```java
byte[] argb = ...
BitSet bits = BitSet.valueOf(argb);
bits.set(0); // sets the 0th bit to true
bits.clear(0); // sets the 0th bit to false
byte[] newArgb = bits.toByteArray();

ByteBuffer bb = ByteBuffer.allocate(4);
bb.put(newArgb);
int i = bb.getInt();
```

https://stackoverflow.com/questions/12100651
You might want to take a look at `BitSet`

```java
byte[] argb = ...;
BitSet bits = BitSet.valueOf(argb);
```

This example throws `BufferUnderflowException`. You must call `ByteBuffer.flip()` to reset the internal buffer.

```java
ByteBuffer bb = ByteBuffer.allocate(4);
//correct: put(newArgb)
int i = bb.getInt();
```

https://stackoverflow.com/questions/12100651
ExampleCheck: Augmenting Stack Overflow with API Usage Patterns Mined from GitHub

try this out, I did not test it, but from what I see in your code, alliances is not an json array, and are json objects based on what I see in your json doc.

0

Just to note, I just tested it and it works fine. – faljbour Apr 25 '15 at 4:45

You may want to check whether match_number.isJsonPrimitive() returns true. You may also want to handle the potential Exception thrown by getAsString() by using a try-catch block here. 48 Github code examples also do this.

```java
try {
    if (match_number.isJsonPrimitive()) {
        match_number.getAsString();
    }
} catch (Exception e) {
}
```

See this in a Github example:
- SpigotMC/BungeeCord
- facebook/buck
- carrotsearch/randomizedtesting
Now available at Chrome Web Store!
ExampleCheck Workflow

Web Browser

Stack Overflow

...<code>API misuse</code>...

Code Extraction

Pop up Generation

ExampleCheck Server

API Misuse Detection

API usage mining on GitHub (offline)
API Usage Mining from GitHub [ICSE 2018]

1. Code Search  Program Slicing  Call Sequence Extraction

380K Java Repositories on GitHub

2. Frequent Sequence Mining

Structured API call sequences

3. SMT-based Guard Condition Mining

API usage patterns
Insight 1: Mining a Large Code Corpus

- Our code corpus includes 380K GitHub projects with at least 100 revisions and 2 contributors.

Insight 2: Removing Irrelevant Statements via Program Slicing

• We perform backward and forward slicing to identify data- and control-dependent statements to an API method of interest.

1. Code Search
2. Frequent Sequence Mining
3. SMT-based Guard Condition Mining

Structured API call sequences

380K Java Repositories on GitHub

API usage patterns
void initInterfaceProperties(String temp, File dDir) {
    if(!temp.equals("props.txt")) {
        log.error("Wrong Template.");
        return;
    }
    // load default properties
    FileInputStream in = new FileInputStream(temp);
    Properties prop = new Properties();
    prop.load(in);
    ... init properties ...
    // write to the property file
    String fPath = dDir.getAbsolutePath()+"/interface.prop";
    File file = new File(fPath);
    if(!file.exists()) {
        file.createNewFile();
    }
    FileOutputStream out = new FileOutputStream(file);
    prop.store(out, null);
    in.close();
}
```java
void initInterfaceProperties(String temp, File dDir) {
    if (!temp.equals("props.txt")) {
        log.error("Wrong Template.");
        return;
    }
    // load default properties
    FileInputStream in = new FileInputStream(temp);
    Properties prop = new Properties();
    prop.load(in);
    ... init properties ...
    // write to the property file
    String fPath = dDir.getAbsolutePath() + "/interface.prop";
    File file = new File(fPath);
    if (!file.exists()) {
        file.createNewFile();
    }
    FileOutputStream out = new FileOutputStream(file);
    prop.store(out, null);
    in.close();
}
```

Data dependency up to one hop, i.e., direct dependency
Insight 3: Capture the Semantics of API Usage

- It is important to capture the temporal ordering, enclosing control structures, and appropriate guard conditions of API calls.

1. Code Search
2. Program Slicing
3. Call Sequence Extraction

Structured API call sequences

Frequent Sequence Mining

SMT-based Guard Condition Mining

API usage patterns
Insight 3: Capture the Semantics of API Usage

```
new File (String); try {; new FileInputStream(File)@arg0.exists(); } catch (IOException) {; }
```
Insight 3: Capture the Semantics of API Usage

```
new File (String); try {; new FileInputStream(File)@arg0.exists(); } catch (IOException) {; }
```
Insight 3: Capture the Semantics of API Usage

```
new File (String); try {; new FileInputStream(File)@arg0.exists(); } catch (IOException) {; }
```

Grammar of Structured Call Sequences

```
sequence ::= ε | call ; sequence
       | construct {; sequence ;} ; sequence
       | call ::= name(t₁, ...tₙ)@condition
construct ::= if | else | loop | try | catch(t) | finally
condition ::= boolean expression
name ::= method name
       | t ::= argument type | exception type | *
```
Insight 3: Capture the Semantics of API Usage

```
new File (String); try {} new FileInputStream(File)@arg0.exists(); } catch (IOException) {};
```
Insight 4: SMT-based Guard Condition Mining

• GitHub developers may write the same predicate in different ways.
Insight 4: SMT-based Guard Condition Mining

- We group guard conditions based on their logic equivalence.
- We use Z3 to prove the logic equivalence of guard conditions.
- \( p \iff q \) is valid iff. \( \neg((\neg p \lor q) \land (p \lor \neg q)) \) is UNSAT.

Two equivalent but syntactically different guard conditions for `substring(int)`: 
\[
\text{arg0} \geq 0 \land \text{arg0} \leq \text{rcv.length()} \iff \text{arg0} - 1 \land \text{arg0} < \text{rcv.length()} + 1
\]
API Misuse Detection

- Contrast SO code snippets with mined API usage patterns automatically.

![Diagram of API Misuse Detection process]

- Stack Overflow snippets
- Call Sequence Extraction
- Structured API call sequences
- Temporal Ordering Check
- Guard Condition Check
- Query pattern(s)
- Patterns
- API usage violations
JsonObject obj =
    root.getAsJsonObject();
JsonElement match_number =
    obj.get("match_number");
...
System.out.println(
    match_number.getAsString());

SO code example [Post 29860000]
JsonObject obj =
    root.getAsJsonObject();
JsonElement match_number =
    obj.get("match_number");
...
System.out.println(
    match_number.getAsString());

getAsJsonObject()@true;
get(String)@true;
...
getAsString()@true;
println(String)@true

Structured Call Sequence

SO code example [Post 29860000]
JsonObject obj =
    root.getAsJsonObject();
JsonElement match_number =
    obj.get("match_number");
...
System.out.println(
    match_number.getAsString());

SO code example [Post 29860000]

getAsJsonObject()@true;
get(String)@true;
...
getAsString()@true;
println(String)@true

Structured Call Sequence
API Usage Pattern for 
JsonElement.getAsString() 

structured call sequence

getAsJsonObject(@true;
get(String)@true;
...
getAsString()@true;
println(String)@true

try {
getAsString()@rcv.isJsonPrimitive();
}
catch (Exception) {
};

An API Usage Pattern of JsonElement.getAsString()
Temporal Ordering Check

Structured Call Sequence

getAsJsonObject()@true;
get(String)@true;
...
getAsString()@true;
println(String)@true

try {
getAsString()@rcv.isJsonPrimitive();
};
catch (Exception) {
};

An API Usage Pattern of JsonElement.getAsString()

missing a try-catch block!
Guard Condition Check

getAsJsonObject()@true;
get(String)@true;
...
getAsString()@true;
println(String)@true

Structured Call Sequence

try {
getAsString()@rcv.isJsonPrimitive();
}
catch (Exception) {
}

An API Usage Pattern of JsonElement.getAsString()

true ⇒ rcv.isJsonPrimitive() and thus incorrect guard condition!
Evaluation Results [ICSE 2018]

- 31% of 217K SO posts contain API usage violations.
- 72% of sampled posts with violations may cause program crashes, resource leaks, etc.
- Highly-voted posts are not necessarily more reliable in terms of correct API usage.
Live Demo
Summary

• Alert users about potential API usage violations in Stack Overflow using patterns mined from 380K GitHub projects

• Expand the scope of APIs beyond 100 Java and Android APIs

• Automate the end-to-end pipeline of API usage mining to keep API usage patterns up-to-date

Tool: https://chrome.google.com/webstore/detail/examplecheck/amliempebckaiaklimcpopomlknklkioe
Dataset: http://web.cs.ucla.edu/~tianyi.zhang/examplecheck.html
Q&A