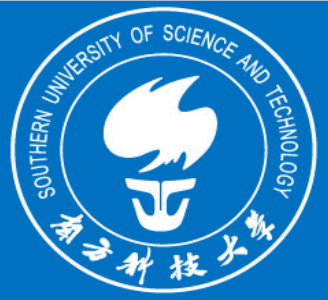


Collaborative bug finding and bug-fixing

Shin Hwei Tan

Southern University of Science and Technology

Shin Hwei Tan

Website： https://www.shinhwei.com/
Email： tansh3@sustech.edu.cn

Research Area

- Automated Program Repair
- Software Testing
- Genetic Improvement

Education Background

UIUC (Bachelor and Master)

National University of Singapore (PhD)

Southern University of Science and Technology (Assistant Professor)

How do Developers Test Android Applications?

"I have not found any easy-to-use testing solutions for Android"

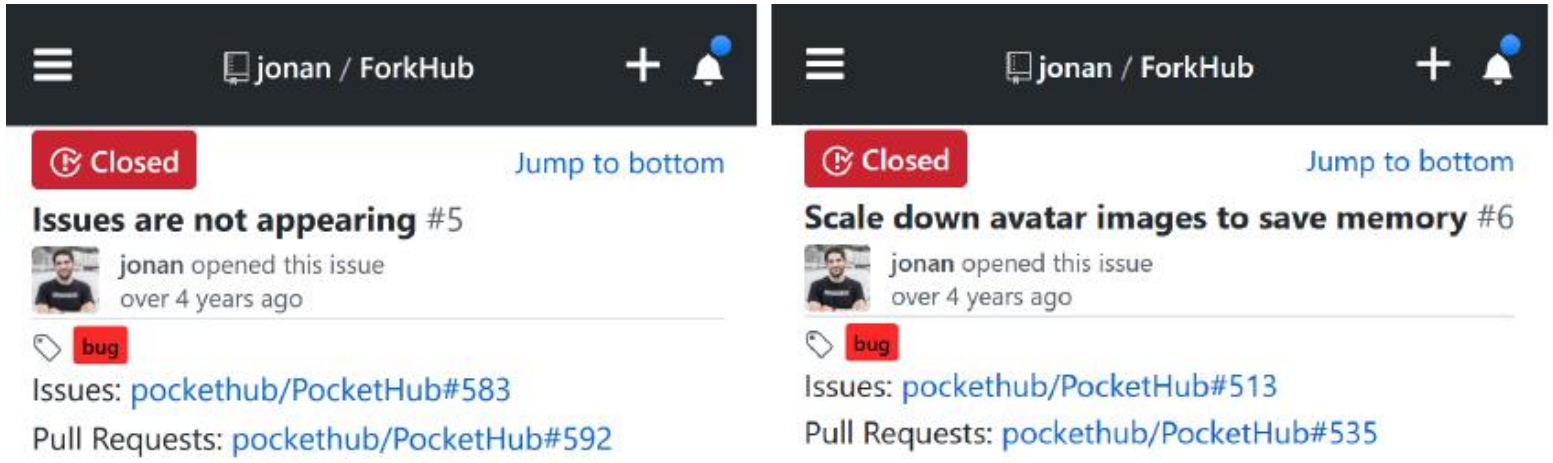
"it's hard to write the useful test case for current project, because the requirement is changed very often, and the schedule is very tiny. So we still prefer hire some tester to do manually testing. In the other hand, the android test framework is not good enough yet, I tried study roboeletric, it's a little bit hard to understand."

"A lot of what I do is related to how the app looks and feels. Therefore a lot of my testing is done manually...

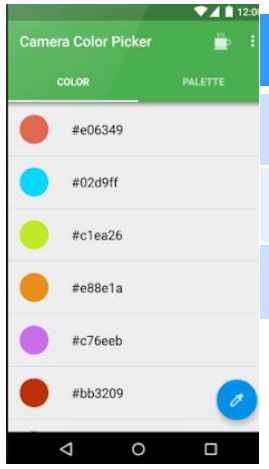
- Developers mostly rely on manual testing and unit testing.
- Developers prefer automatically generated test cases in natural language

from "*How do Developers Test Android Applications?*" [ICSME'17]

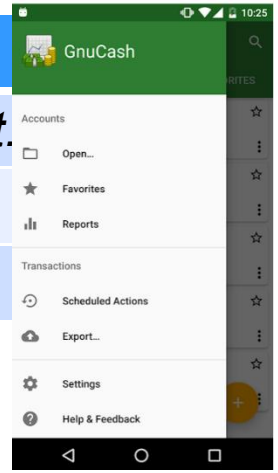
Many bug reports exist in repositories like GitHub



App developers who developed PocketHub and ForkHub found similar bugs across two apps

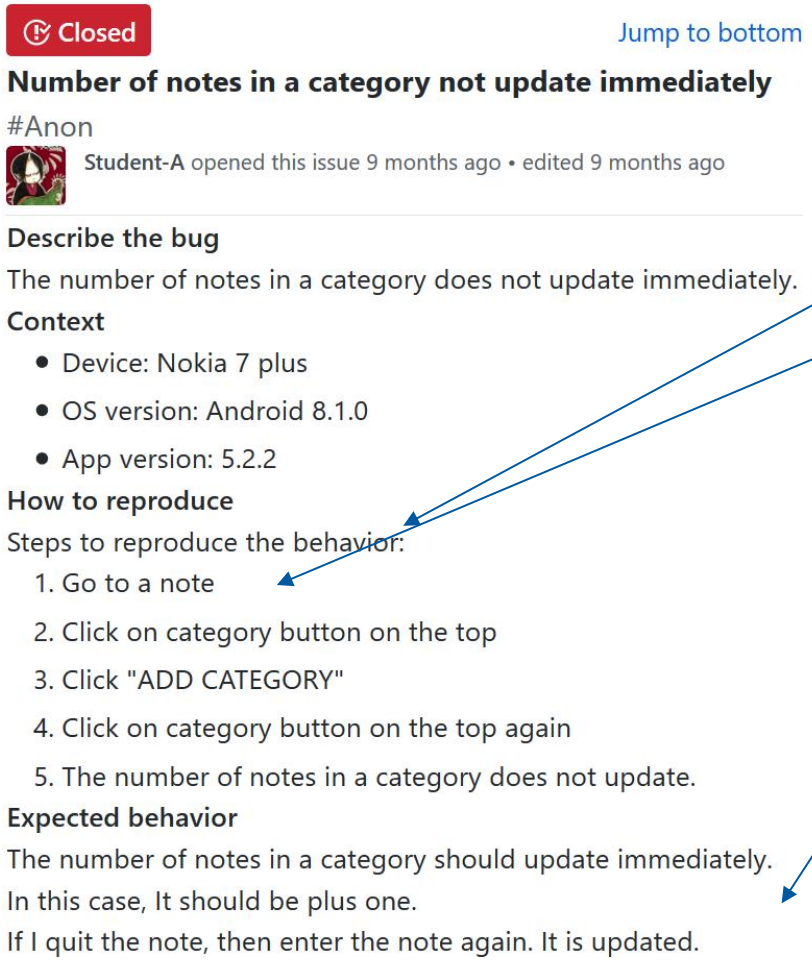


Bug report's title in CameraColorPicker	Bug report's title in Gnucash
How do I get left top color.	*When an account is edited, its color is lost.*
publish to F-Droid	*Publishing on F-Droid*
Make gradlew executable	*Make gradlew executable*



There exists one-to-one correspondence for the bug report's title in apps of different categories!

Crafting test scenario from bug report



Closed

Jump to bottom

Number of notes in a category not update immediately

#Anon

Student-A opened this issue 9 months ago • edited 9 months ago

Describe the bug

The number of notes in a category does not update immediately.

Context

- Device: Nokia 7 plus
- OS version: Android 8.1.0
- App version: 5.2.2

How to reproduce

Steps to reproduce the behavior:

1. Go to a note
2. Click on category button on the top
3. Click "ADD CATEGORY"
4. Click on category button on the top again
5. The number of notes in a category does not update.

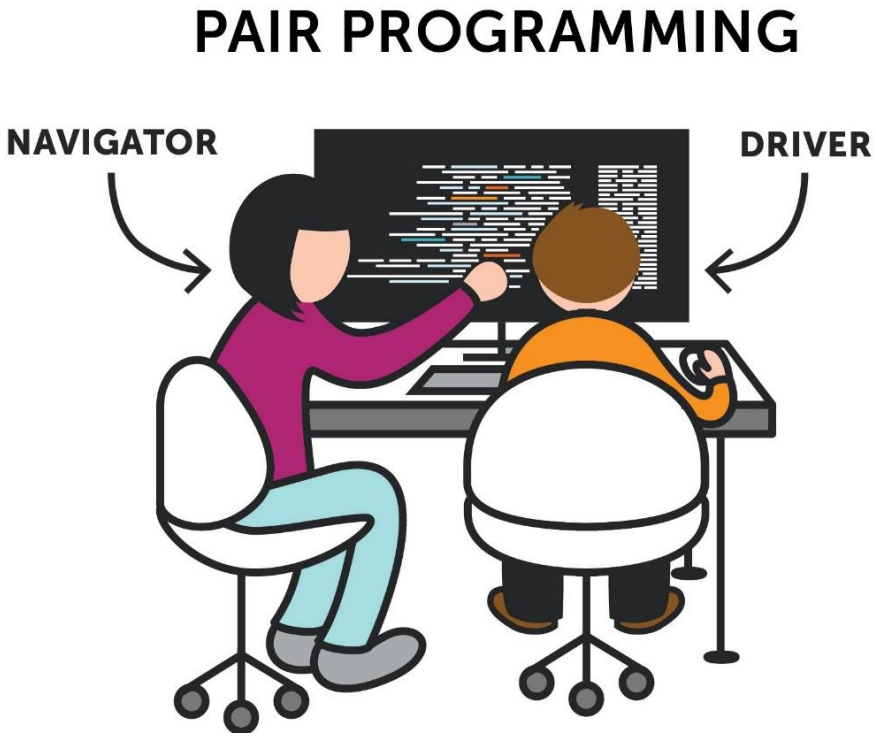
Expected behavior

The number of notes in a category should update immediately.
In this case, It should be plus one.
If I quit the note, then enter the note again. It is updated.

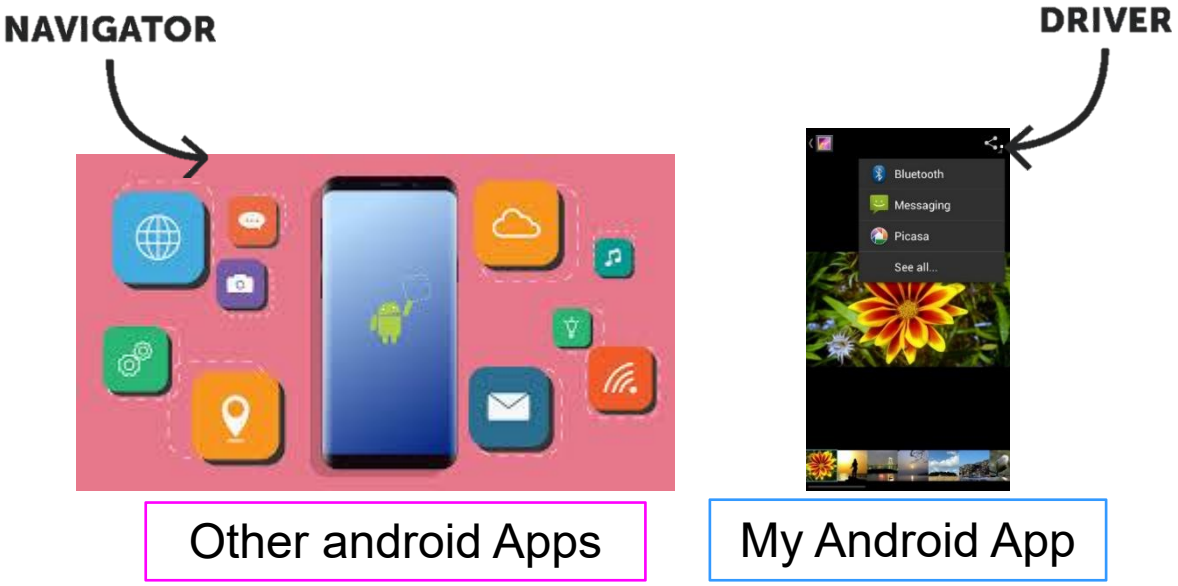
- A test scenario includes:
  - steps to reproduce
  - test data (e.g., an image for image processing app)
  - the expected behavior
    - Could solve the test oracle problem

How does collaborative bug finding works?

- Emulate the role of a competent pair programmer via developers of other similar applications

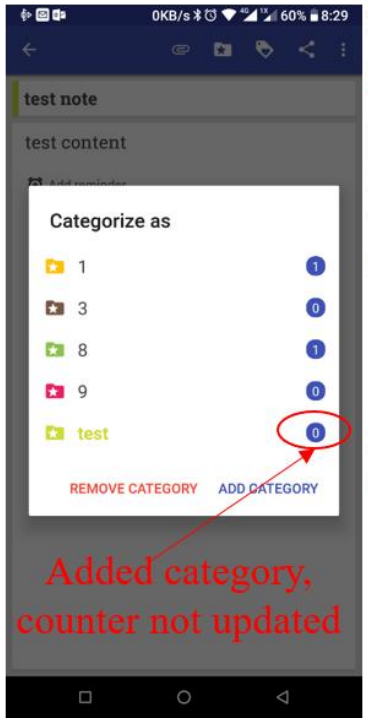


- Designed 3 settings to model different interactions between coders
  - Coders-vs-Coders
  - Coder-vs-Manual-Issues
  - Coder-vs-Auto-Issues (Bugine)



Motivating Example

i: original shared issue



Closed — Jump to bottom

Number of notes in a category not update immediately #Anon

Student-A opened this issue 9 months ago • edited 9 months ago

Describe the bug

The number of notes in a category does not update immediately.

Context

- Device: Nokia 7 plus
- OS version: Android 8.1.0
- App version: 5.2.2

How to reproduce

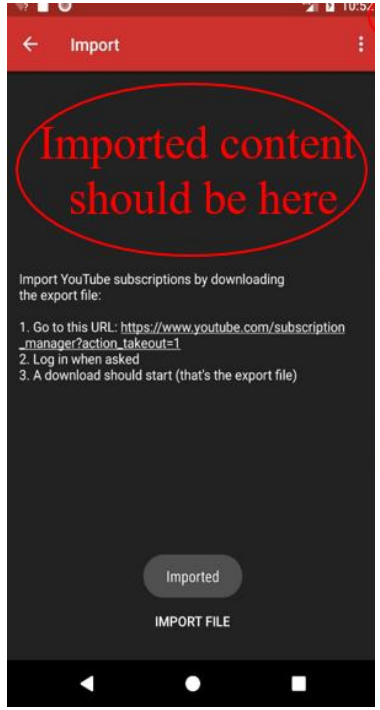
Steps to reproduce the behavior:

1. Go to a note
2. Click on category button on the top
3. Click "ADD CATEGORY"
4. Click on category button on the top again
5. The number of notes in a category does not update.

Expected behavior

The number of notes in a category should update immediately.
In this case, It should be plus one.
If I quit the note, then enter the note again. It is updated.

j: derived issue.



Open — Jump to bottom

Import subscriptions from YouTube not update immediately #Anon

Student-B opened this issue 9 months ago • edited 9 months ago

Describe the bug

After import Subscriptions from YouTube by a subscription_manager file, the Subscriptions page don't update immediately.

Context

- Device: Nexus 5X
- OS version: Android 8.0.0
- App version: newest dev (1/12/2018)

How to reporduce?

Steps to reproduce the behavior:

1. Go to Subscriptions
2. Select import from Youtube
3. Go to the URL and download a file
4. Select file and choice the file download
5. The app show a toast "imported"

Expected behavior

The app should update this page immediately. Only If I go to the home page and reopen the subscriptions menu, it will be updated.

1 1 1 +

Student-E referenced this issue from another issue

#Anon Import from previous export and update strangely

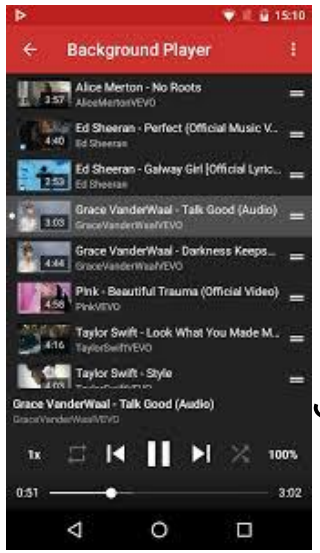
View are not immediately update, will only update the view after restarting the app

- Prevalent problems for many apps
  - 5 pair sharings

Setting 2: Coder-vs-Manual-Issues

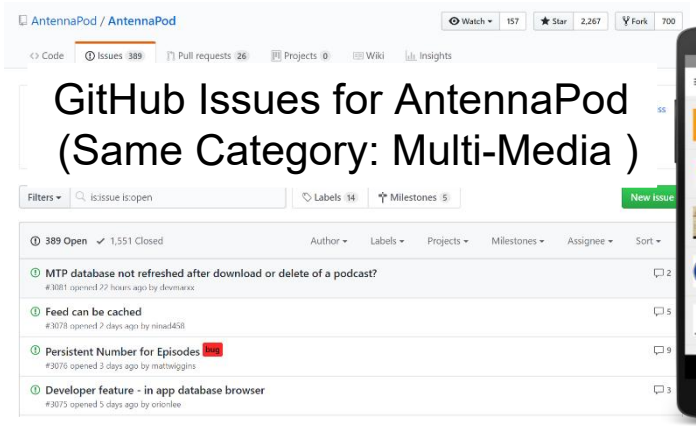
DRIVER

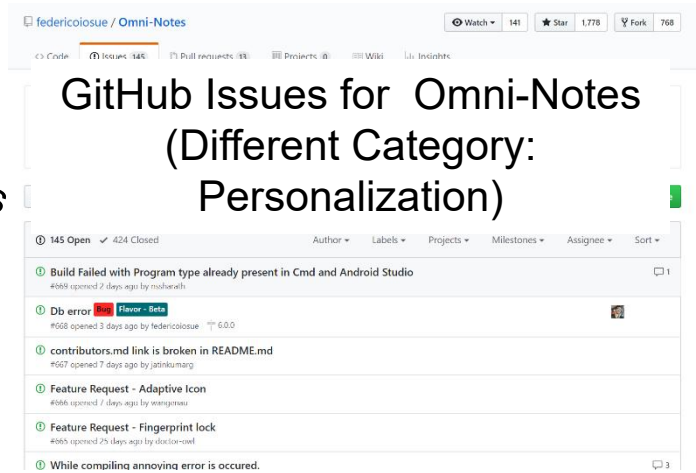
Developer for NewPipe

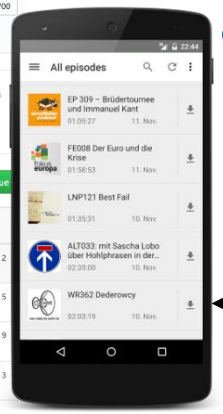


Select 5 issues

Select 5 issues



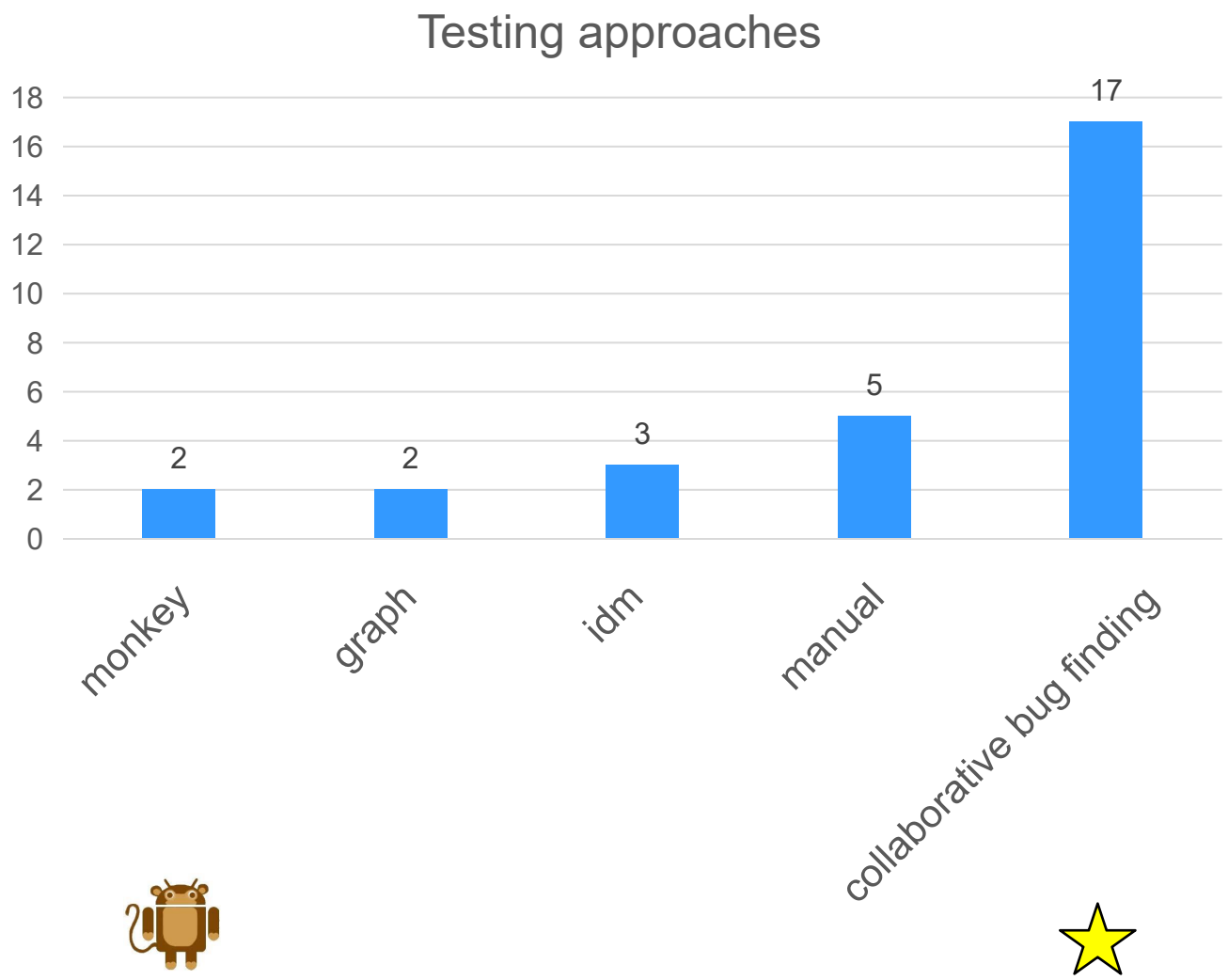




- Coder A needs to manually perform the steps below:
  1. Select the relevant issue *i*
  2. Reproduce the steps in *i*
  3. Check if the same bug described in *i* applies for the app by Coder A

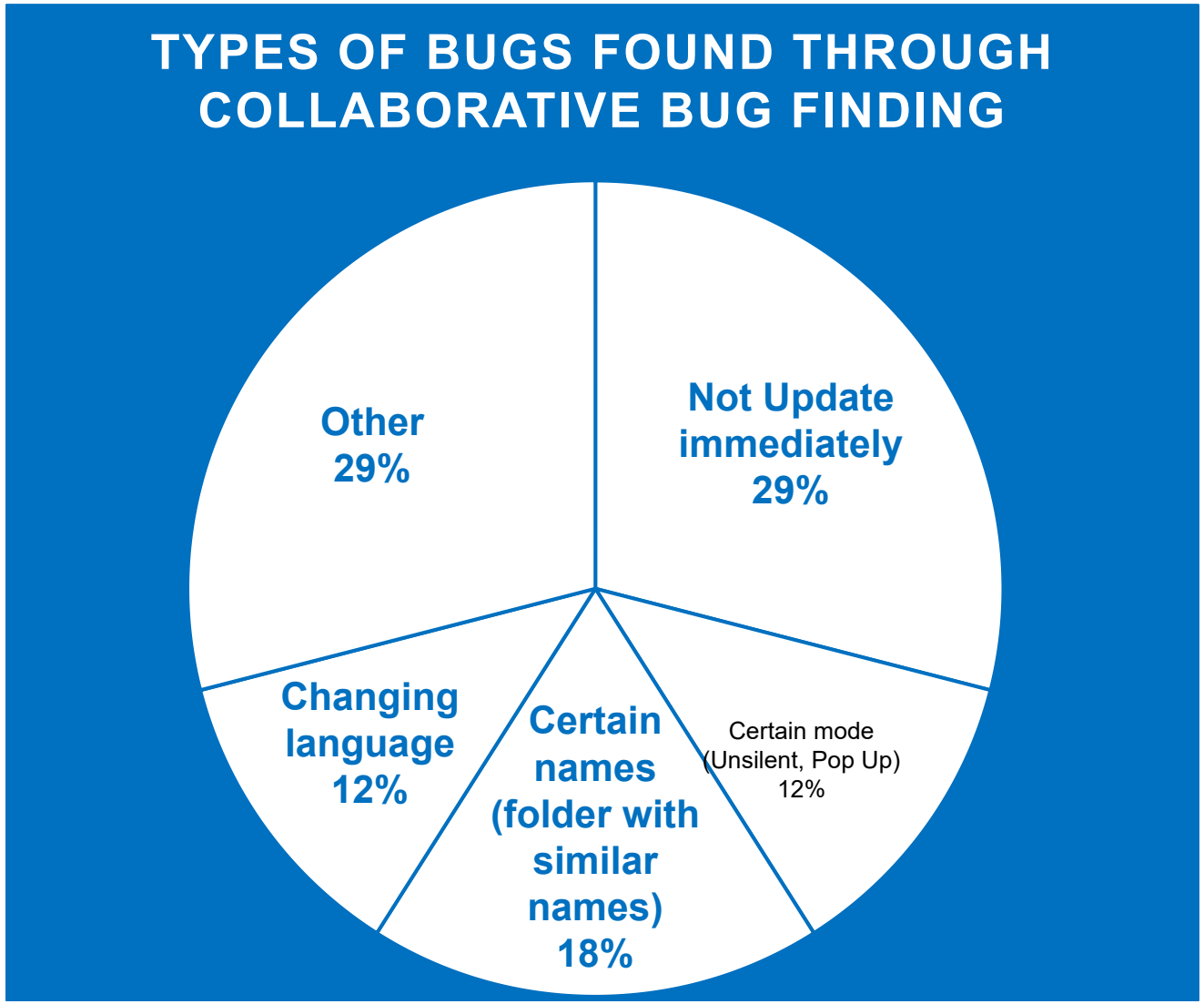
NAVIGATORS

Effectiveness of these two settings



- Collaborative bug finding finds 17 new bugs
- Automatic testing tool like Monkey only find 2 bugs
  - Crashes found are hard to reproduce

Types of Bugs found



- Our approach finds:
  - Prevalent problems (outdated view, certain names, certain mode, change of language)
  - Specific problems (29%)
- Types of bugs are mostly non-crash related
  - Our approach complements existing automated testing approaches (mostly focus on finding crashes).

What do students think about collaborative bug finding?

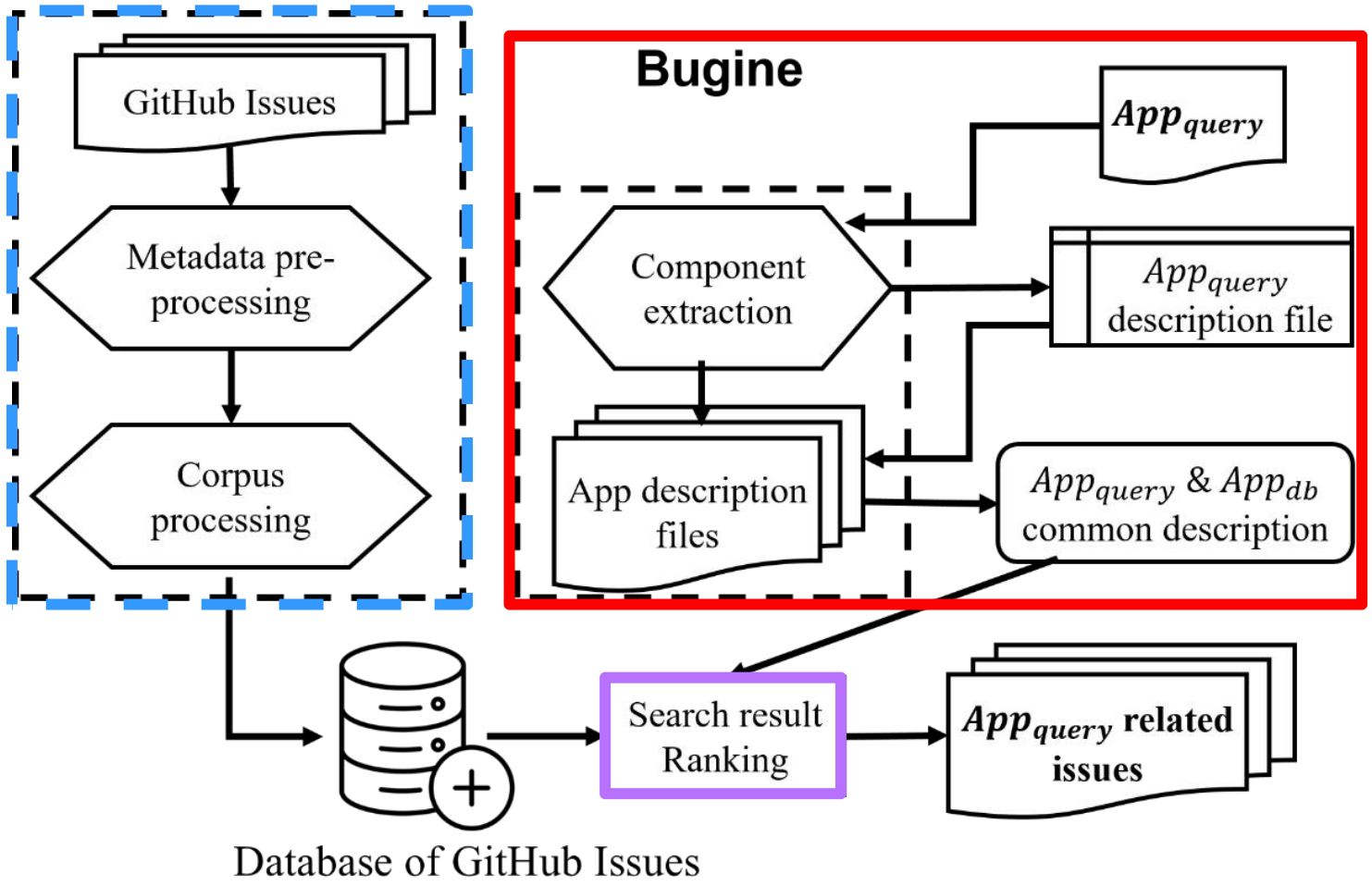
"Because many functions in the app in the same category are similar even totally same ... and others' report will also inspire the mind to find bugs which I never considered."

"Collaborative bug finding takes more time to review different apps and search useful issues...but is more likely to find new bug"

Searching for issues could be time-consuming

- Need automation for collaborative bug finding!

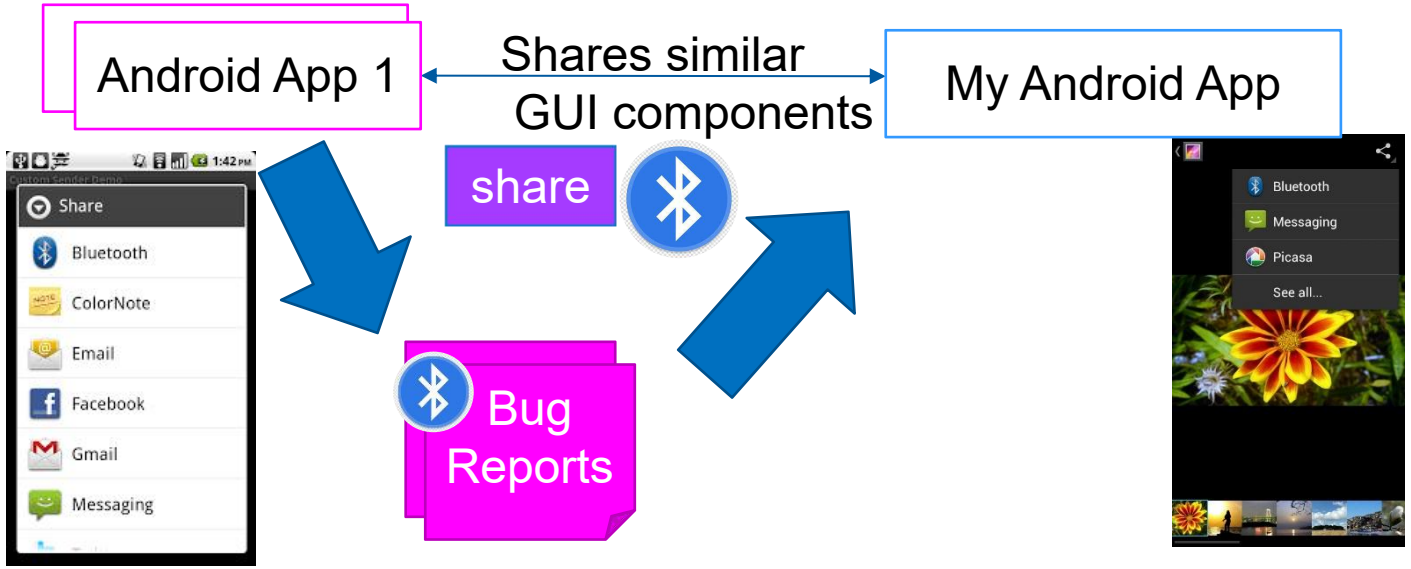
Setting 3: Coder-vs-Auto-Issues (Bugine)



- GitHub issues pre-processsing
- Given an App_{query},

1. Extracts its UI components to get its app description file
2. Use similarities between app description file for App_{query} and app description files for apps in database to search for similar apps
3. Rank issues based on quality

Finding the similarities between apps

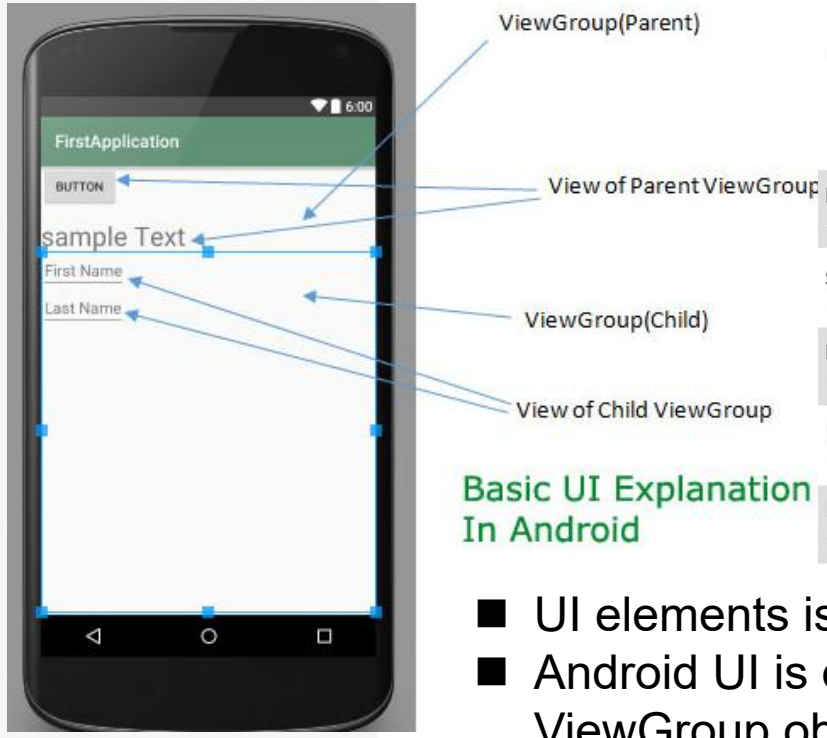


Has some bug reports that mentioned the "shared" GUI components, recommend these bug reports to my app

Are these two apps similar?

Background: What defines Android UI?

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
 android:orientation="vertical" android:layout_width="match_parent"
 android:layout_height="match_parent">


 <Button
 android:id="@+id/buton1"
 android:layout_width="wrap_content"
 android:layout_height="wrap_content"
 android:text="Button"/>

 <TextView
 android:id="@+id/textView1"
 android:layout_width="wrap_content"
 android:layout_height="wrap_content"
 android:text="sample Text"
 android:layout_marginTop="15dp"
 android:textSize="30dp"/>

 <RelativeLayout
 android:layout_width="match_parent"
 android:layout_height="match_parent">

 <EditText
 android:id="@+id/editTextName"
 android:layout_width="wrap_content"
 android:layout_height="wrap_content"
 android:hint="First Name"
 />

 <EditText
 android:id="@+id/editTextLastName"
 android:layout_width="wrap_content"
 android:layout_height="wrap_content"
 android:hint="Last Name"/>

 </RelativeLayout>
</LinearLayout>
```



ANDROID RESOURCE NAMING CHEAT SHEET

by @MOLSJEROEN

<WHAT>_<WHERE>_<DESCRIPTION>_<SIZE>

fixed set of options, choose the right one below | custom part Android view subclass, "all" if reused in ≠ screens | differentiate multiple elements in one screen | always optional, [xdp] or bucket [small]

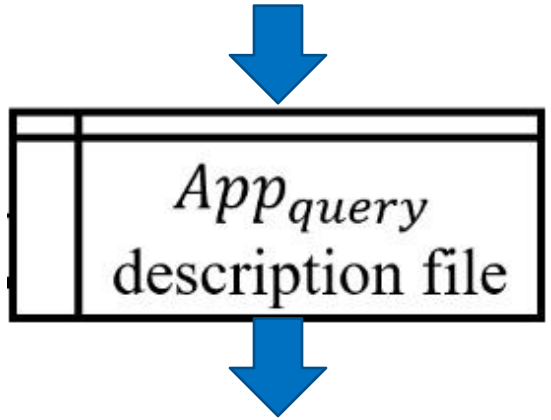
LAYOUTS <WHAT>_<WHERE>.XML	<WHAT> is activity, fragment, view, item or layout e.g. activity_main.xml
STRINGS <WHERE>_<DESCRIPTION>	e.g. main_intro all_done
DRAWABLES <WHERE>_<DESCRIPTION>_<SIZE>	e.g. all_infoicon_small main_background
IDS <WHAT>_<WHERE>_<DESCRIPTION>	<WHAT> is name of Android/Custom view class e.g. linearlayout_main_fragmentcontainer
DIMENSIONS <WHAT>_<WHERE>_<DESCRIPTION>_<SIZE>	<WHAT> is width, height, size, margin, padding, elevation, keyline or textsize e.g. keyline_all_text

- UI elements is usually declared in XML files
- Android UI is defined via the hierarchy of View and ViewGroup objects
- Names of Android resource following certain conventions

From： https://abhiandroid.com/ui/xml

Naming information for extracting description files

Component	Description	Example	Extracted Names
Resource Name	Resource Name	android:id="@+id/my_btn"	my_btn
View Name	Name for the type of UI component	<Button android:id="@+id/my_btn" />	Button
XML File Names	Layout name	main_layout.xml	main_layout

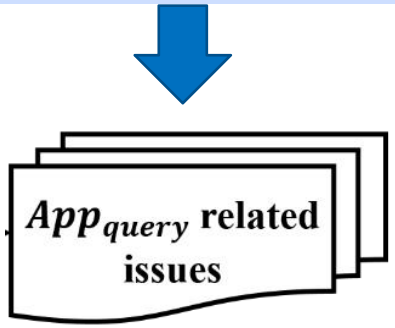


For each XML file, convert each view and each resource in App_{query} to the query of the form:

XML file name $\wedge$ View Name $\wedge$ Resource name

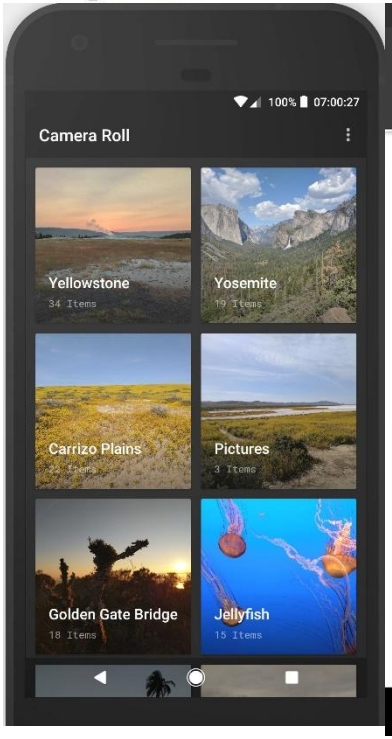
Ranking GitHub Issues

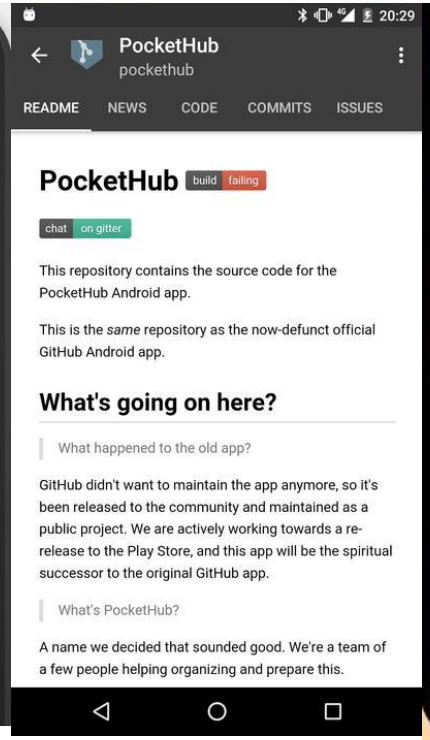
Factors	Description	Rationale
Issue length	Word count of issue body (int)	Longer issue is better
Issue Status	Closed or opened (binary)	Issue is more important if it is : ✓ Closed ✓ Fixed ✓ has more replies (comments)
Ref Commit SHA	Commit SHA referenced by issue (binary)	
Issue Reply Number	The number of replies that an issue received (int)	
Hit_all	Find all search keywords in the corpus (binary)	Search for shared UI components: ✓ Match all keywords (better) ✓ Has some shared components
Hit_overlap	Overlap coefficient between search keywords and corpus (float)	
Hit_Hot_Words	Word count of descriptive hot words like reproduce, defect (int)	Issues that meet the criteria for good bug reports is better

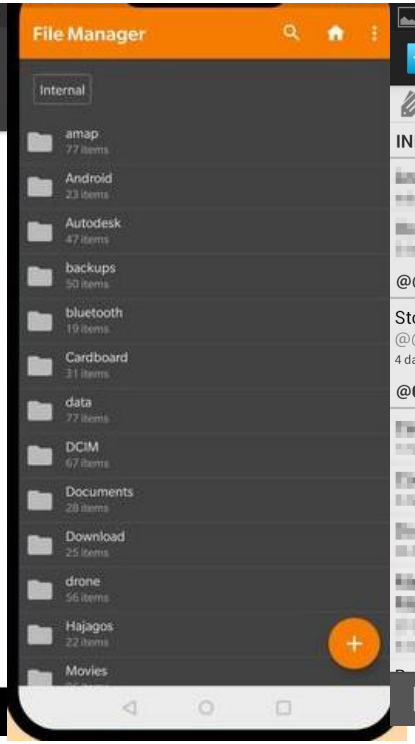


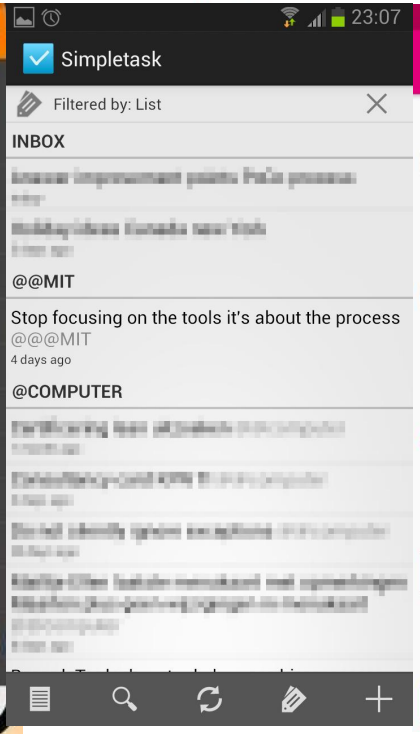
Evaluation of Bugine

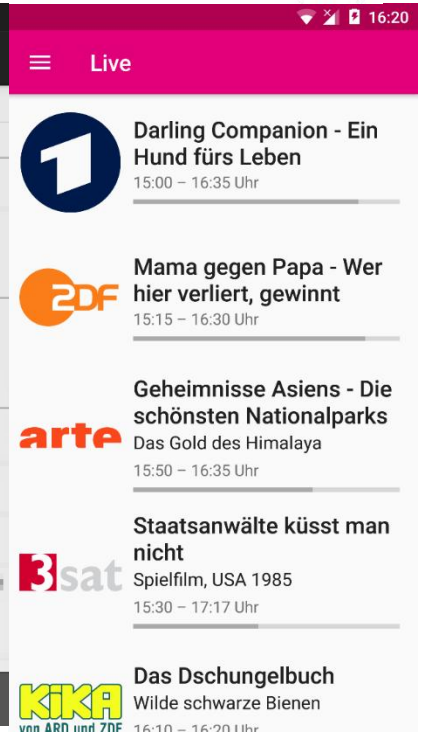
App Name	Category	KLOC	#Downloads	Rating	Version No.	#GitHub Stars	#GitHub Issue (closed)
Camera-Roll	Gallery	26.00	100,000+	4.2	1.0.6	420	227(133)
PocketHub	GitHub client	31.35	10,000+	3.3	0.5.1	9429	644(526)
Simple File Manager	Explorer	5.84	50,000+	4.5	6.3.4	378	189(130)
Zapp	Broadcast	8.41	N.A.	N.A.	3.2.0	60	151(137)
Simpletask	Reminder	24.80	10,000+	4.7	10.3.0	349	821(583)











Evaluate the ranking performance of Bugine

Use two metrics commonly used in prior recommendation systems:

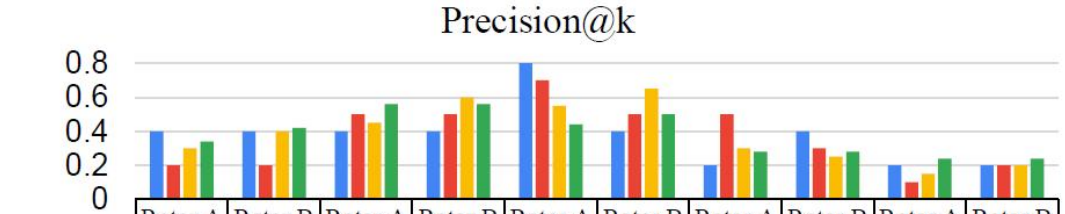
$$(1)\ \text{Prec@k} = \frac{\#\ relevant\ documents\ in\ top\ k}{k}$$

- Retrieval precision over the top k documents in the ranked list

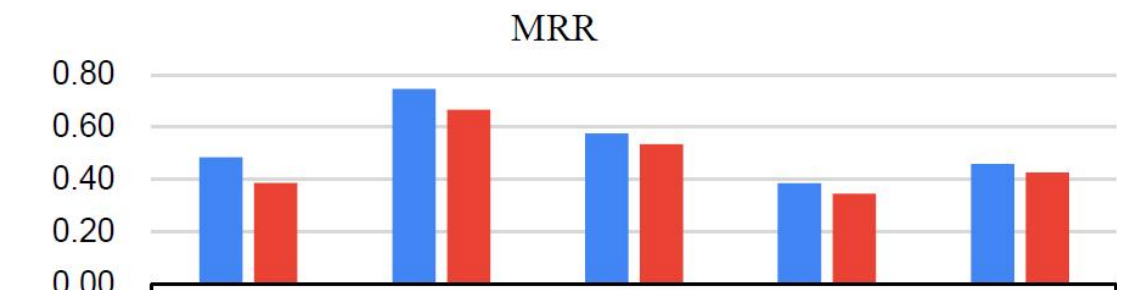
$$(2)\ \text{Mean Reciprocal Rank (MRR)} = \frac{1}{|Q|}\sum_{q=1}^{|Q|}\frac{1}{first_q}$$

- For each query q, the MRR measures the position $first_q$ of the first relevant document in the ranked list

Ranking performance of Bugine



	Rater A	Rater B	Rater A	Rater B	Rater A	Rater B	Rater A	Rater B	Rater A	Rater B
	Camera-Roll		PocketHub		Simple File Manager		zapp		Simpletask	
Prec@5	0.4	0.4	0.4	0.4	0.8	0.4	0.2	0.4	0.2	0.2
Prec@10	0.2	0.2	0.5	0.5	0.7	0.5	0.5	0.3	0.1	0.2
Prec@20	0.3	0.4	0.45	0.6	0.55	0.65	0.3	0.25	0.15	0.2
Prec@50	0.34	0.42	0.56	0.56	0.44	0.5	0.28	0.28	0.24	0.24



	Camera-Roll	PocketHub	Simple File Manager	zapp	Simpletask
Rater A	0.48	0.75	0.58	0.39	0.46
Rater B	0.39	0.67	0.53	0.34	0.43

- Prec@10: 0.1–0.7
  - Among the top 10 issues recommended by Bugine, there is ≥ 1 relevant issue
- MRR values: 0.34–0.75
  - Ranking for the first relevant document is btw 3rd (0.34) and 1st (0.75)

➢ Bugine could recommend relevant issues for all evaluated apps

Bugs found by Bugine

App Name	#Bugs Found (new, old)
Camera-Roll	(11, 0)
PocketHub	(12, 2)
Simple File Manager	(6, 2)
Zapp	(2, 7)
Simpletask	(3, 2)

- Found 34 new bugs and 13 old bugs
  - In first two settings, 29 students find 17 new bugs in 20 apps
  - Bugine could discover more bugs despite being evaluated only on five apps.

Feedbacks from the ICSE 2020 Reviewers

"This is one of those simple but great ideas that make a lot of sense…"

"I thought the idea of collaborative testing was intriguing and thought provoking… I really liked the effort by the authors to think creatively about this and present an out of the box idea for test generation"

"The idea of collaborative bug finding is refreshing and interesting"

Interesting Research Questions:

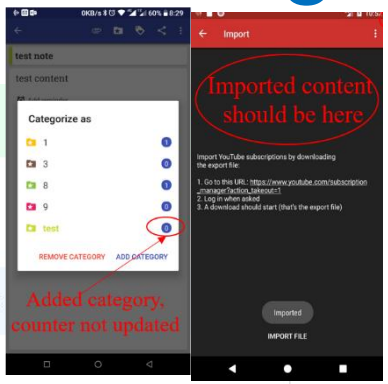
Can the authors provide any insights on how to automate collaborative bug finding?

- Can we extract fix patterns from these common issues?

Collaborative Bug Fixing

Could we make sure of the fixes of similar bugs for Automated Program Repair?

```
+ import rx.Observable;
+ import rx.functions.Func1;

  import static com.nineoldandroids.view.ViewPropertyAnimator.animate;
  import static java.lang.Integer.parseInt;

@@ -1216,10 +1218,15 @@ private void toggleChecklist2(final boolean keepChecked, final boolean showCheck
         * Categorize note choosing from a list of previously created categories
         */
        private void categorizeNote() {
-               // Retrieves all available categories
-               final ArrayList<Category> categories = DbHelper.getInstance().getCategories();

                String currentCategory = noteTmp.getCategory() != null ? String.valueOf(noteTmp.getCategory().getId()) :
+         final List<Category> categories = Observable.from(DbHelper.getInstance().getCategories()).map(category -> {
+             if (String.valueOf(category.getId()).equals(currentCategory)) {
+                 category.setCount(category.getCount() + 1);
+             }
+             return category;
+         }).toList().toBlocking().single();
+
```



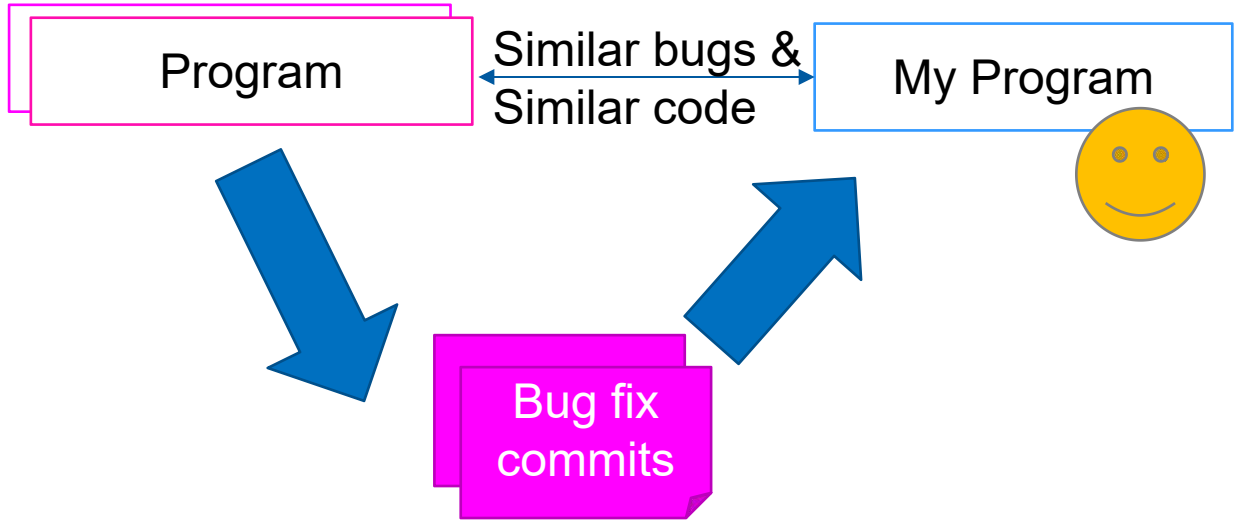
mauriciocolli commented on 25 Apr 2019 • edited ▾ Collaborat

- Sort the items chronologically and by type (live, then upload date in descending order)
- Let users group their subscriptions
- Load the feed items in parallel in a background service (like importing works)
- Show the watching count in live stream items
- Use `Groupie` library for easier development of lists

- Could provide high-level fix suggestion
  - Use Observable vs. Background Service for handling asynchronous events for fixing the outdated view problem

Future Work: Collaborative Testing and Repair

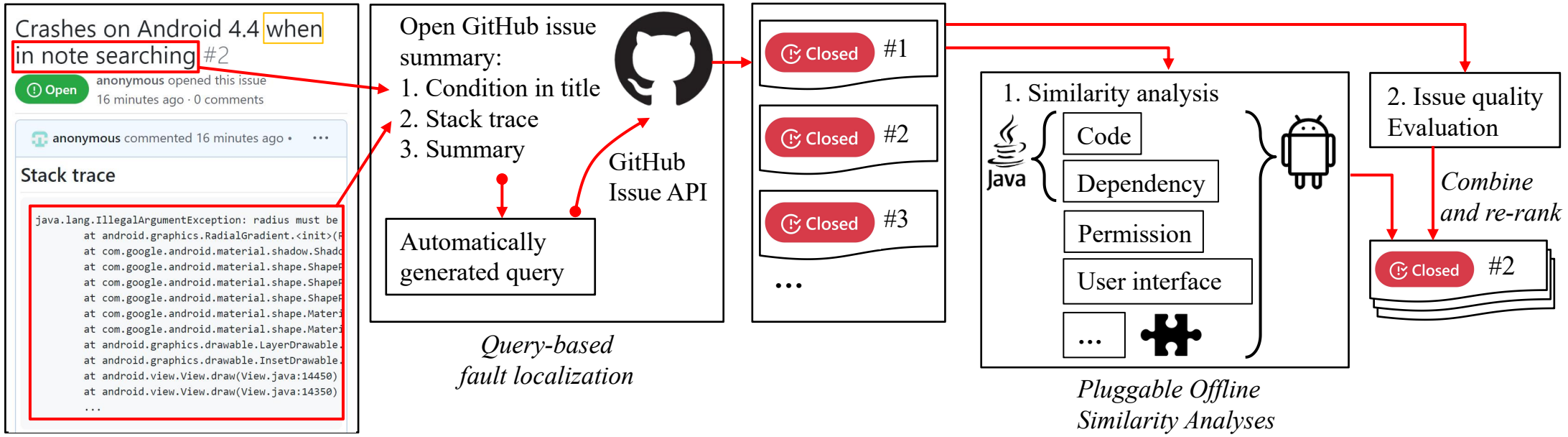
- How about recommendation systems for bug fix commits?



Has bug fix commits, recommend these commits to another Program

CrossFix: Resolution of GitHub Issue via Recommendation of Similar Bugs

- Similarity measurements
  - Code
  - Dependency
  - Permission
  - UI
- More general: Can be used for any Java applications beyond Android apps



SUSTech Southern University of Science and Technology

GitHub-OSS Fixit: Fixing bugs at scale in a Software Engineering Course


Shin Hwei Tan


Chunfeng Hu


Ziqiang Li


Xiaowen Zhang


Ying Zhou

Approaches that use OSS

SUSTech Southern University of Science and Technology

Approach	Repository Selection	Issues Selection	Participants	Mentors
GSoC	Limited to organizations that have signed up for GSoC	Limited to the listed projects that implement a feature	Selected students that have signed up for GSoc	A few designated mentors from each organization
FindBugs fixit	Limited to FindBugs	Limited to issues in FindBugs	Google engineers who participated in the “fixit”	User who have reported the bug
GitHub-OSS fixit 	Flexible. Can select any project	Flexible. Can select issues that fix bugs	Students in a class	The person who reported the GitHub issue & a developer assigned to the issue who will approve pull requests.

GitHub-OSS Fixit

SUSTech Southern University of Science and Technology

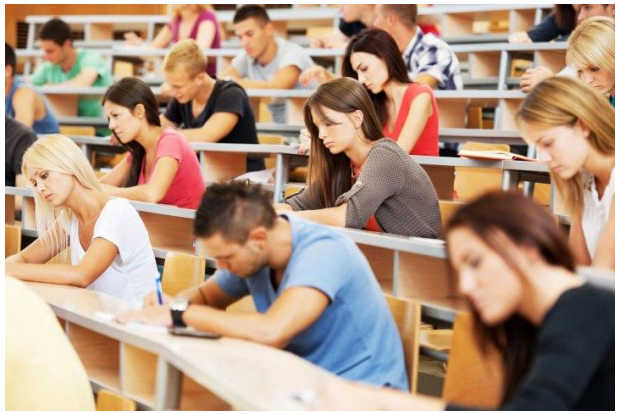
- Emulate the role of a mentor via developer of open-source projects who opened a GitHub issue
- Key ideas: Students fix bugs in open-source projects in GitHub

GITHUB-OSS FIXIT

MENTORS

Other Open-source software (OSS)

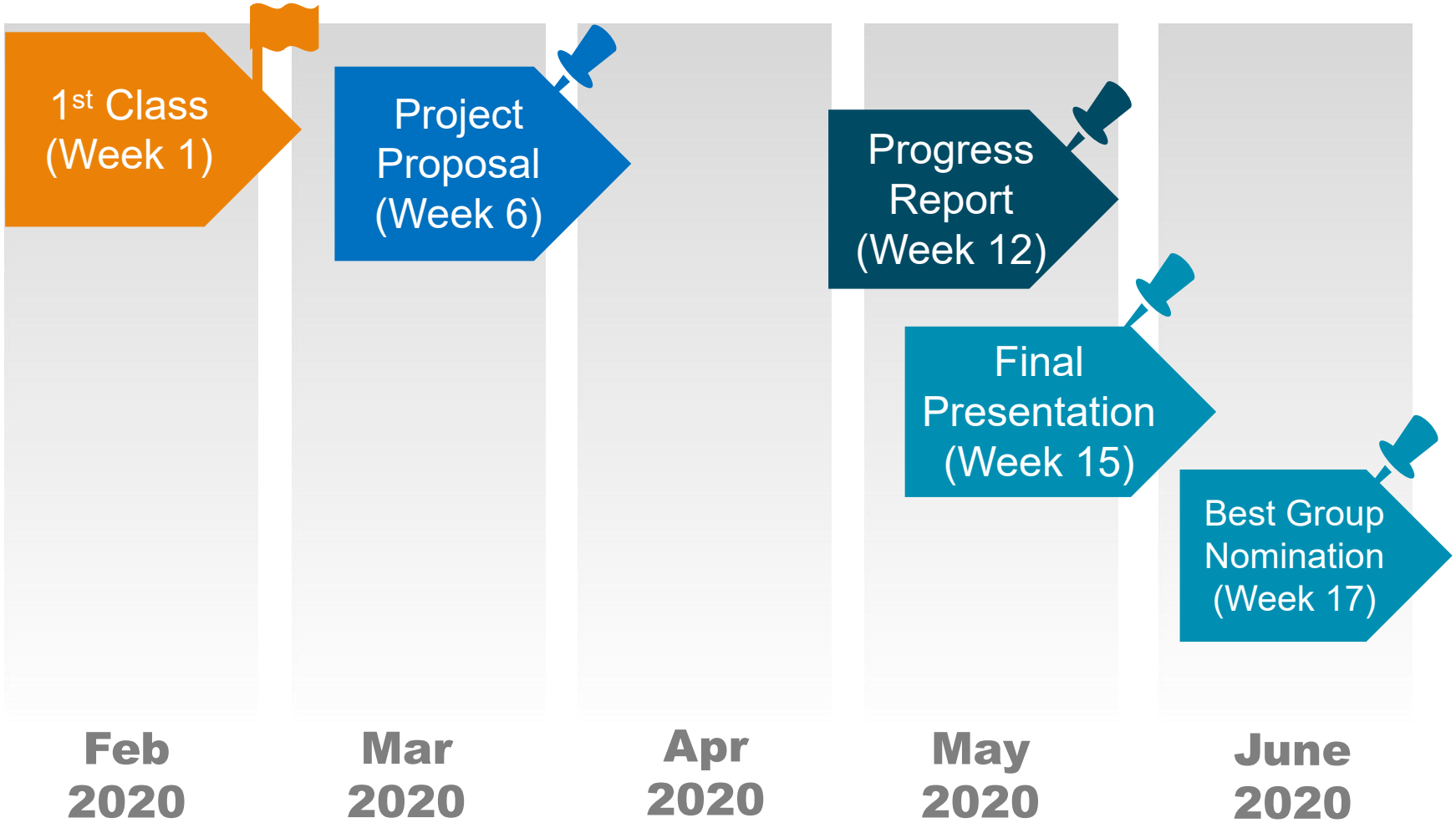
DRIVER



Students

Timeline for GitHub-OSS Fixit

SUSTech Southern University of Science and Technology



Instructions

- ***How to select projects and issues?***

Project Proposal: Issue Selection Criteria

SUSTech Southern University of Science and Technology

Importance of the issues: Issue should be important. For example, important issues have tags like “up for grab” or “help wanted”. For most open source projects (e.g, INRIA/spoon), you could find the issues that are suitable for beginners under the “contribute” link (e.g, for INRIA/spoon, the “contribute” link is https://github.com/aaa/spoon/contribute.

No fixing commit: Issue should not have any fixing commit. Do not select any issue if a contributor has volunteered to fix it. This is a compulsory criterion.

Reproducible: Make sure that the issue is reproducible and you will need to write a test case for reproducing the issue (this test case could be a system-level test or test included in the issue). Do not select an issue if you do not understand the issue and cannot reproduce it. Do not select an issue if someone mentioned in the comment that it is not reproducible. This is a compulsory criterion.

Estimated lines of code to resolve issue: Selected issue should be non-trivial to resolve. The estimated lines of code needed to resolve issue should be greater than adding/modifying 10 lines of code.

Estimated time to resolve issue: Estimated time taken to resolve one selected issue should be at least 1-2 weeks instead of 1-2 days.

Project Proposal: Selecting GitHub issues

SUSTech Southern University of Science and Technology

Select 5-20 issues to fix. Each issue needs to be selected by following the "**guidelines for selecting issues"**. The number of selected issues depends on (1) the number of group members, (2) the difficulty of the selected issues, and (3) the estimated time to fix the issue (there may be some related issues that could be fixed together so you will need to adjust the estimated accordingly). Note that you need to consider the time taken in understanding the project and the issue (1 week for learning is allowed for each issue). The total estimated time for the group members should be at least 6 weeks per person. This means that if you have *x* team members in your group and the total estimated time for fixing all issues should be 6 multiplied by x (Total=6*x*). For example, if an issue takes 3 weeks to fix, then each person could choose to fix 2 issues for the entire project. **Please plan your issues accordingly, fixing those issues that are more likely to be fixed by others first.** For all the selected issues, create a table such as below: *(8 points)*

Link to issues	Type of issues (Bug/Feature)	Estimated Time to fix each issue	Number of people for fixing this issue (≤2 members for each issue)	Estimated Difficulty (in a scale of 1-5, 1 means very easy to fix and 5 means very difficult to fix)
https://github.com/INRIA/spoon/issues/3118	Feature	3 weeks	1	5
…				
Total: 50 weeks				

Project Proposal: Roles for team member

SUSTech Southern University of Science and Technology

The team members should be diverse with each member playing different roles in a software development team. Include a division of roles for each team member for the first iteration of the project. Note that for each iteration, the member will get to take up different roles. *(5 points)*

Name	Role
	Leader
	Developer
	Developer
	Designer (Contribution guidelines & code review)
	Tester
	Developer & Documentation (JavaDoc)

The results

We were blown away by the creativity displayed by the educators in bringing Computer Science to life! From across the world, we saw educators that teach at different levels of education share their insights. From HTML, to algorithms, hardware and Git, you can get all sorts of inspiration from our winning submissions. Congratulations to Rahul, Shin Hwei, Juan Alberto, Shivangi and Kimberly!

You can check out the full lesson plans in TwilioQuest's **Awesome-CS repository** on GitHub.

The GitHub Fixit Project

This 6-week lesson plan, aimed at **higher-ed** students, helps students boost their employability by contributing to well-known open source projects. The course covers core software engineering concepts including static analysis, coding standards, unit testing, and the very important skill of making pull requests.

Lesson Plan

- ***Won World Teacher Day Challenge 2020!***
  - ***Co-organized by GitHub***

Lesson Plan

SUSTech Southern University of Science and Technology

Week	Lecture	Lab
1	Why learn SE?	Roles in Software Development Teams
2	Version Control & Build System	Git and GitHub
3	Waterfall model & eXtreme Programming (XP) - Pair Programming	GitHub Classroom and Markdown
4	Planning Game & Unit Testing	Unit testing with JUnit
5	TDD & Code Coverage	Test coverage and Automated test generation with Evosuite
6	Mutation Testing & Evosuite	PIT and Measuring Mutation Coverage
7	Reverse Engineering	Metrics in Java Projects
8	-	-
9	Software Metrics	Static Analysis Tools (PMD, Checkstyle, and Findbugs)
10	Static Analysis	Reverse Engineering and Testing Android Apps
11	Defensive Programming & Documentation	Javadoc and Doxygen
12	-	-
13	Software Reuse & Component-based Software Engineering	Code Review for Progress Report
14	UI Design	Progress Report Presentation
15	DevOps and Continuous Integration	Common vulnerabilities in Java programs
16	Security Engineering	Exam Review

- ✓ Taught multiple automated tools during lab sessions to ensure code quality and reduce software maintenance effort:
  - JUnit, Evosuite, PIT, Metrics Plugin, PMD, Checkstyle, Findbugs, Doxygen

Peer Tutoring

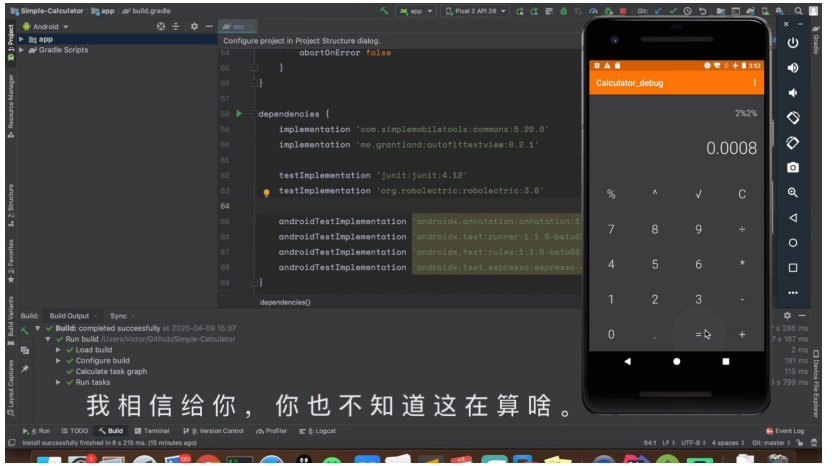
Asked previous students on “How to start from 0 involving in open source project and resolve their issue?”

- Conduct Interview via Instant Messaging
- Video by senior student

How to start from 0 involving in open source project and resolve their issue?

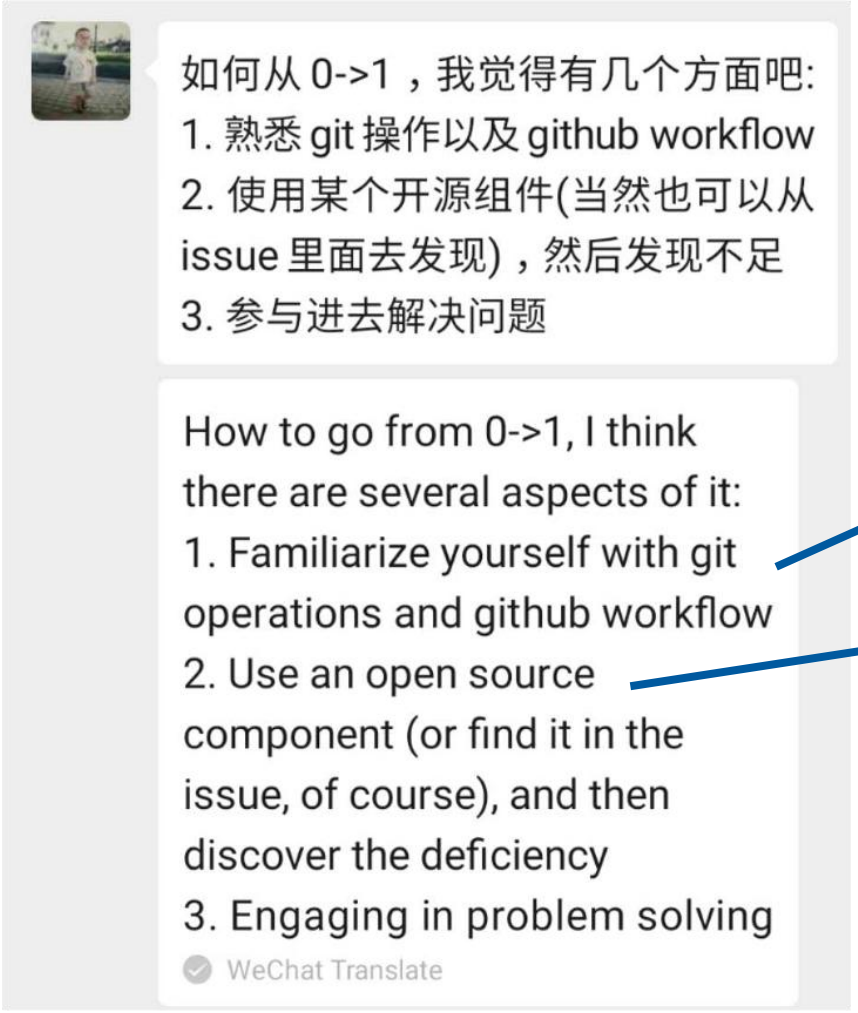
SUSTech Southern University of Science and Technology

- Previous student who took SE and ST (Yifei Xu) recorded a video
  - https://www.bilibili.com/video/BV1dc411h7VH previously contributed to OSS in GitHub with an accepted pull request
  - In the software testing class in previous semester, he has gone through all the steps (starting from 0 in the software testing class)!
    1. Find bugs in open-source Android apps
    2. Prepare test case for the bug (Test-driven development)
    3. File a GitHub issue: https://github.com/SimpleMobileTools/Simple-Calculator/issues/139
    4. Fix the bug by creating pull request
    5. His pull request has been recently accepted by the developer!



How to start from 0 involving in open source project and resolve their issue?

SUSTech Southern University of Science and Technology



- Conducted an Interview with previous SE student helper (ZeHuai Wang)
  - previously contributed to OSS in GitHub (Kubernetes) with an accepted pull request

✔ Taught Git and github workflow in one of the lab and students use git and Github Classroom for homework submission!

- Encourage students to use the selected open source project/libraries
  - Students could find bugs in them and issue the pull request for the bug that you found! (Starting from finding bugs is actually starting from 0! They are currently starting from a GitHub issue so they already have some information about the bugs)

Results

Experimental Setup

154 students in class

140 students participated

1. What are the benefits and disadvantages of GitHub-OSS Fixit based on students' feedback?
2. Do students perceive GitHub-OSS Fixit as a good course project?
3. How many GitHub issues can the students resolve during the GitHub-OSS Fixit project?
4. What is the effectiveness of the additional resources provided by senior students?

Benefits of GitHub-OSS Fixit

Benefits	# of responses
Learn XP practices (coding standards, static analysis)	71
Teamwork	57
Read others' code	55
Practical project (can contribute to open-source)	54
Learn GitHub and Git	50
Improve skills (team communication, problem-solving)	45
Realistic (learn about structure of big project)	25
Inherit the benefits of Git & GitHub (efficient)	24
Good experience (build confidence, fun, innovative)	21
Communicate with developer	13

Survey results

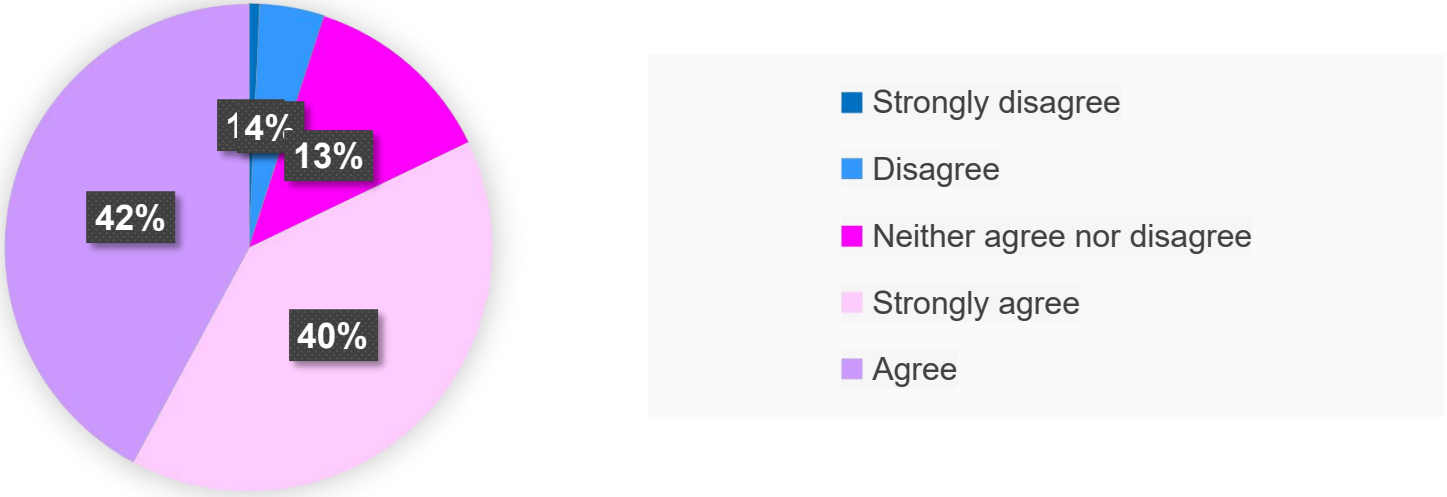
154 students

214 pull requests (PR)

93 PRs are merged

46 PRs are closed

Overall, I would recommend GitHub-OSS Fixit as a project



- Overall, most students agreed that GitHub “fixit” is a good project for a Software Engineering course and would recommend it.

Impact of GitHub-OSS Fixit

Invited China Computer Federation (CCF) article

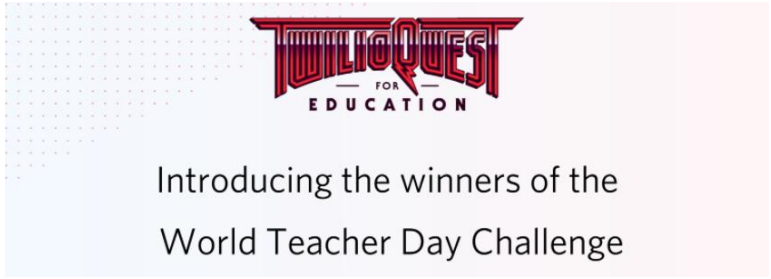
数字图书馆 Digital Library

首页 文章 视频 音频 图片 PPT 专辑

将开源和企业引入计算机课程教学——"把教学场景用起来"模式探讨

本科教育中的实践与创新

Our lesson plan won World Teacher Day Challenge



TwilioQuest FOR EDUCATION

Introducing the winners of the World Teacher Day Challenge

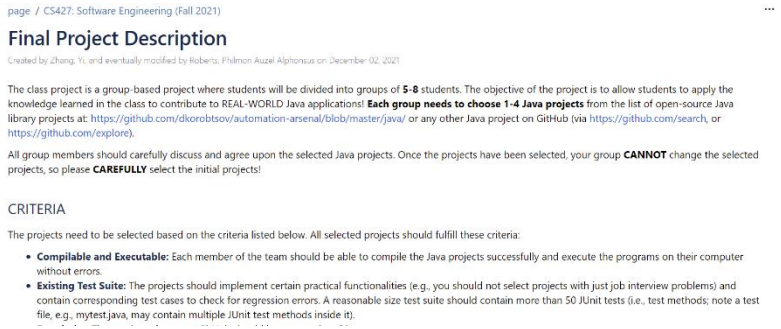
We were blown away by the creativity displayed by the educators in bringing Computer Science to life! From across the world, we saw educators that teach at different levels of education share their insights. From HTML, to algorithms, hardware and Git, you can get all sorts of inspiration from our winning submissions. Congratulations to Rahul, Shin Hwei, Juan Alberto, Shivangi and Kimberly!

You can check out the full lesson plans in TwilioQuest's **Awesome-CS repository** on GitHub.

The GitHub Fixit Project

This 6-week lesson plan, aimed at **higher-ed** students, helps students boost their employability by contributing to well-known open source projects. The course covers core software engineering concepts including static analysis, coding standards, unit testing, and the very important skill of making pull requests.

Project adapted by CS427 Fall 2021 in UIUC



page / CS427: Software Engineering (Fall 2021)

Final Project Description

The class project is a group-based project where students will be divided into groups of **5-8** students. The objective of the project is to allow students to apply the knowledge learned in the class to contribute to REAL-WORLD Java applications! **Each group needs to choose 1-4 Java projects** from the list of open-source Java library projects at: https://github.com/dkorobtsov/automation-arsenal/blob/master/java/ or any other Java project on GitHub (via https://github.com/search, or https://github.com/explore).

All group members should carefully discuss and agree upon the selected Java projects. Once the projects have been selected, your group **CANNOT** change the selected projects, so please **CAREFULLY** select the initial projects!

CRITERIA

The projects need to be selected based on the criteria listed below. All selected projects should fulfill these criteria:

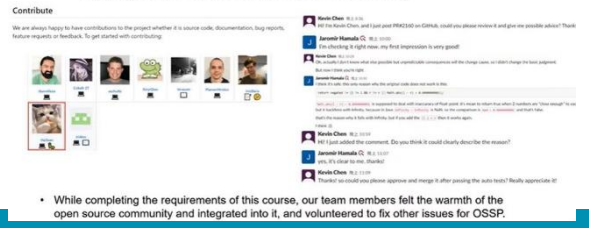
- **Compilable and Executable:** Each member of the team should be able to compile the Java projects successfully and execute the programs on their computer without errors.
- **Existing Test Suite:** The projects should implement certain practical functionalities (e.g., you should not select projects with just job interview problems) and contain corresponding test cases to check for regression errors. A reasonable size test suite should contain more than 50 JUnit tests (i.e., test methods; note a test file, e.g., mytest.java, may contain multiple JUnit test methods inside it).

Local Companies expressed interests in collaboration

I am Liu Xukun (Student No.: 11912823) who chose your "Software Engineering(CS304)" course this semester. I'm emailing you because the person in charge of the project "DataEase" we selected this semester appreciates your design of the course project that encourages everyone to work together for the open source project. At the same time, the general manager of their company is also very interested in this. Would you like me to ask you whether there is the possibility of school enterprise cooperation?

The following is their company's website https://www.fit2cloud.com/ , the branch is located in Futian, Shenzhen.

- Enjoying the atmosphere in open source community



Contribute

- While completing the requirements of this course, our team members felt the warmth of the open source community and integrated into it, and volunteered to fix other issues for OSSP.

Students invited to be contributors of OSS!

Website

- Checkout our website for more information
  https://github-fixit.github.io/

Future Research

Automated Program Repair

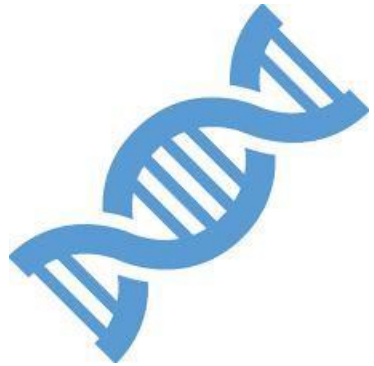
Applications:

- Crashes of Android Apps [ICSE'18]
- Synchronization bugs in GPU programs [ASE'19]
- Linux Kernel Patch backporting [ISSTA'21], Won Distinguished Artifacts Award
- Web UI Test Repair (Huawei Grant)
- Feedback generation for programming assignments [FSE'17]
- Autogenerated programs by Codex

Software Testing

Find bugs in:

- Comment-Code Inconsistencies [ICST'12], Won David J. Kuck Outstanding MS Thesis Award
- Android Apps [ASE'18]
- Conformance Testing for JavaScript Engine [PLDI'21]
- Static Analysis Tools



Search-based Software Engineering

Applying search-based techniques in:

- Search-based Automated Program Repair [ICSE'15, FSE'16,ICSE'18]
- Search-based Automated Test Generation [ICST'12]