World Teacher Day Challenge 2020

Lesson Title: The GitHub Fixit project

Level: Beginner/Intermediate/[✓ Advanced]

Lesson Duration: 6 weeks

Compatible with distance learning? [✓ Yes]/No

Lesson Objectives:

- Learn about Git and GitHub, particularly focus on reading GitHub issues and making pull requests
- Provide an opportunity for students to make real contributions to open source community in GitHub
- Provide a platform for students to learn software engineering concepts, including static analysis, coding standard, unit testing
- Help students to boost their CVs through participation in well-known open source projects

Technology Requirements/ Equipment:

Students: Computer, GitHub classroom, Blackboard

Educator: Computer, GitHub classroom, Blackboard

Lesson Description (As much detail as necessary):

Background: Since 2005, Google Summer of Code (GSoC) has demonstrated success in introducing students into open source software development by assigning the participants a programming project with an open source organization during their break [1]. GSoC program offers many benefits, including (1) allows students to get involved in open source projects, (2) allows students to practice Java programming skills, (3) provides opportunities to communicate with the mentors from open source projects, etc. al. Meanwhile, Google had successfully conducted a company wide "*fixit*" where many of their engineers reviewed thousands of warnings, and fixed bugs in FindBugs (a static analysis tool) [2].

Why GitHub "fixit"?: Inspired by the success of GSoC and the FindBugs "fixit", we conducted a GitHub "fixit" where students select their favorite open source Java projects and fixed bugs reported in GitHub in these projects. For each reported bug, there will be a person who filed the bug report and he or she could serve as a virtual "mentor" who will provide students with the instructions required (e.g., the expected behavior) to reproduce the bug. Compared to GSoC, our GitHub fixit project is more flexible because GSoC participants are required to register during a certain period of time and could only work with one open source project that they have signed up for. Meanwhile, we adapted the idea of fixing bugs from the FindBugs "fixit" project and brought it to life in a classroom setting. Hence, it is easier for teachers to incorporate this as part of their semester-long course project. Table 1 below shows the differences between our approaches and other existing approaches.

Table 1: The differences between GitHub "fixit" and other approaches

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 "fixit"	Flexible. Can select any project	Flexible. Can select issues that fix bugs/implement a feature	Students in a class	The person who reported the GitHub issue. An issue will have a developer assigned to the issue

				who is in charge of approving pull requests.

Computer Science Concepts Covered:

- Version Control System (Git)
- GitHub (GitHub issues and making pull requests)
- Unit testing
- System/Integration tests
- Javadoc comments
- Coding standard
- Test-driven development (TDD)
- Java programming
- Static Analysis

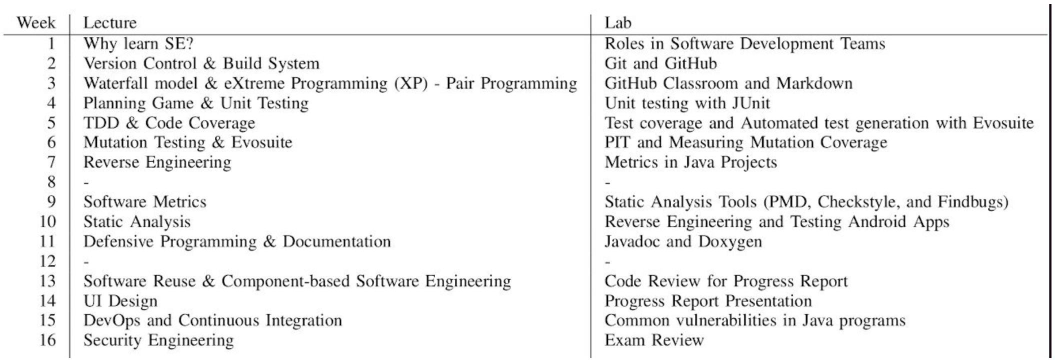
Level of education: Junior/Senior students taking Software Engineering course

Overview: The GitHub fixit project consists of three major stages: (1) project proposal, (2) progress report, and (3) final presentation. To encourage active participation, we announced at the start of the course project that we will select one best group where each member will receive 5 bonus points for the team that made the greatest contributions to open source projects. In each stage, we provided detailed instructions in each stage to help students to achieve the ultimate goal of contributing to open source projects while writing good quality. In the project proposal, students (in a group of 5–6) chose 1–2 Java open source project and the corresponding GitHub issues (bug reports) to work on. In the progress report stage, they presented the code and tests that they have written, and performed a code review with the teaching assistant via online virtual meetings. In the final presentation stage, they summarize their work completed within the semester and finalize the code and tests to be submitted to the corresponding open source project. Below shows the timeline for the GitHub fixit project together with the corresponding link:
Week 1–3: Project Proposal
Week 4–7: Progress Report
Week 8–10: Final Presentation

Week 11: Best group video (Video by the team that was voted as the best group

We also carefully designed the lecture and the lab sessions to cover the related topics for providing basic knowledge that students need for participating in the GitHub “fixit” projects. Table 2 shows the weekly schedule of the entire semester. Particularly, our lab sessions focuses on teaching students automated tools (e.g., Git, GitHub, JUnit, Evosuite, PIT, PMD, Checkstyle, Findbugs, etc al.) that will be useful for ensuring the quality of the written code.

Table 2: Weekly schedule and topics covered during the lectures and the lab sessions



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

Integrated Distance learning: The instructions for all stages are uploaded online in Blackboard [3], a learning management system. Meanwhile, as part of their assignments, students submit their code to the repository created and managed in GitHub classroom [4]. All answers to the questions asked in the instructions are submitted to GitHub classroom as a “README.md” file written according to the GitHub Markdown syntax.

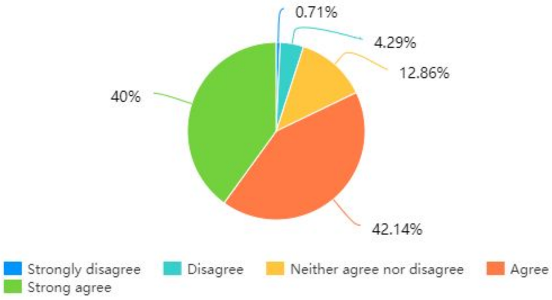
Accessible to all students? The project is centered around GitHub, the largest site for hosting open source projects with over 28 million public repositories, and with the aim of encouraging users to contribute to open source projects. Hence, our project is designed according to the spirit of free and open source software which aims to make software accessible to all users. With this goal in mind, our project is also accessible to all students, with the only requirement being the ability to connect to GitHub to participate in the project. In fact, due to the COVID-19 pandemic, all students have participated in the project when collaborating with their teammates at home. All teaching (including lectures and lab sessions) and meetings are conducted virtually online. The only restriction is that all teaching materials and the teaching itself are conducted in English. In terms of diversity, although most students are students from

China, there is one international student from Cambodia. Moreover, the main instructor is from Malaysia.

Outcomes: We evaluate the effectiveness of the GitHub “fixit” projects by analyzing both qualitative and quantitative data.

To assess the outcome qualitatively, we request for students’ feedback by creating an online survey. Questions asked in the survey are publicly available at: https://github-fixit.github.io/survey-question.pdf.

Among the 154 students taking the course, 140 students participated in the survey. Overall, most students agreed that GitHub “fixit” is a good project for the course and would recommend it. The figure below shows the collected responses for the question “Overall, I would recommend GitHub fixit for the class project.”.



As shown in the figure, most students either agree (42.14%) or strongly agree (40%) that they will recommend GitHub “fixit” for the class projects. The average score for the five-point Likert scale (1 denotes strongly disagree and 5 means strongly agree) is 4.16, which means that most students like the GitHub “fixit” project. In terms of knowledge gained, we asked students if they have improved their skills on the relevant concepts taught in the GitHub “fixit” project. Table 3 shows the gathered responses for the question on the skills that students have improved during the project. Overall, students have responded positively to the effectiveness of GitHub “fixit” by agreeing that they have improved their skills on most of the Software Engineering knowledge covered in class.

Table 3: Results for the question “During the GitHub fixit, I have improved my skills on:”

Knowledge	Strong Disagree	Disagree	Neutral	Agree	Strong Agree
Version Control System (Git) :	1.43%	0.71%	7.86%	50.71%	39.29%

GitHub :	1.43%	0.71%	4.29%	42.86%	50.71%
Unit test:	1.43%	0.71%	5.71%	49.29%	42.86%
System/Integration tests:	2.86%	7.14%	32.86%	36.43%	20.71%
Code comment (Javadoc):	1.43%	2.14%	11.43%	50%	35%
Coding standard:	2.14%	0%	15%	45%	37.86%
Test-driven development (TDD):	2.14%	4.29%	27.14%	41.43%	25%
Java programming	2.14%	1.43%	14.29%	52.14%	30%
Static Analysis Tools:	1.43%	0.71%	8.57%	42.86%	46.43%
Working in a team:	1.43%	5.71%	15%	44.29%	33.57%

We also collected students' opinions about the benefits and disadvantages of GitHub "fixit". As shown in the table below, students listed many benefits of GitHub "fixit", including the ability to learn about some key practices of eXtreme Programming (XP), learn about teamwork, and reading others' code, etc. al. Although we expect GitHub "fixit" to be beneficial in terms of gaining knowledge about XP and Git, we are pleasantly surprised that some students think that GitHub "fixit" is challenging and fun.

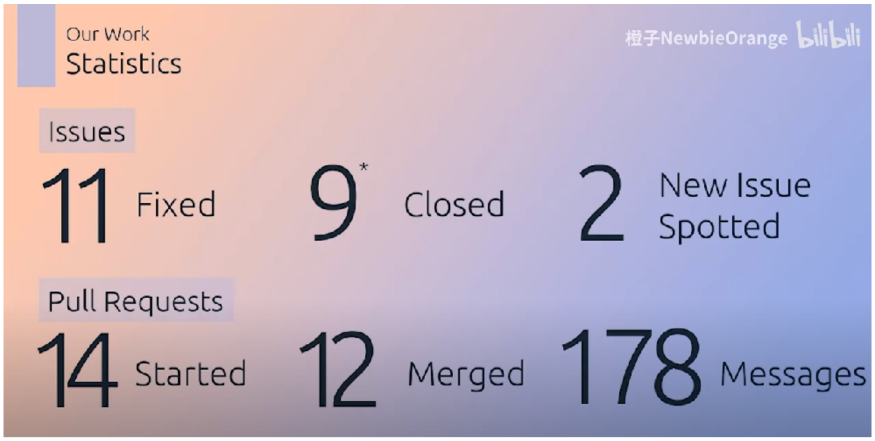
Benefits	# of responses
Learn XP (coding standard, static analysis)	71
Learn teamwork	57
Read others' code	55
Practical project (can contribute to open-source)	54
Learn GitHub and Git	50

Improve skills (communication, coding)	45
Realistic (learn about structure of big project)	25
Obtain benefits of Git & GitHub (efficient)	24
Personal experience (confident, challenging, fun)	21
Communicate with Developer	13
N/A	6

To gather quantitative data, we analyze the pull requests created by the students during the GitHub "fixit" projects. Below is the summary of the results:

- **# of students:** *154* students
- **# of contributed Java projects:** *25* different open source projects
- **# of submitted pull requests:** *214* pull requests to fix bugs across
- **# merged pull requests:** *59*
- **# closed pull requests:** *82*

The results show that 154 students have submitted 214 pull requests, which means that students have submitted on average 1.39 pull requests per person. Moreover, the submitted pull requests have received positive feedback from developers considering the fact that 82 of the submitted pull requests have been accepted by the developers. Consider the result for the best team (team JOJO) voted by the students (as shown in the figure below). In a team consisting of 5 students, students from the team JOJO have successfully started 14 pull requests with 12 of them being merged by the Picocli [5] developers.



Our Work
Statistics

橙子NewbieOrange bilibili

Issues

11 Fixed
9* Closed
2 New Issue Spotted

Pull Requests

14 Started
12 Merged
178 Messages

World Teacher Day Challenge 2020

Based on the qualitative and quantitative data discussed above, we believe that the GitHub “fixit” is successful project because (1) students have given positive feedbacks regarding the usefulness of the project, and (2) it makes real contribution to open source projects, which is far beyond what is taught in the classroom setting. On top of that, the COVID-19 pandemic introduces new challenges where all teaching and meetings need to be held virtually online. Despite the difficulties, students are still able to collaborate well and are able to make real contributions to open source projects.

Homework (if applicable):
Below is the list of assignments and their links:
Week 1–3: Project Proposal
Week 4–7: Progress Report
Week 8–10: Final Presentation

References:

1. https://summerofcode.withgoogle.com/archive/
2. Ayewah, N., & Pugh, W. (2010, July). “The google findbugs fixit”. In *Proceedings of the 19th international symposium on Software testing and analysis* (pp. 241-252).
3. Nagel, David (July 13, 2011). "Blackboard Learn Expands Open Education Standards Support". In THEJournal. Retrieved February 10, 2014.
4. https://classroom.github.com/
5. https://github.com/remkop/picocli

Additional notes: