Teaching a Testing Concept (JUnit) with Active Learning

Kesina Baral
Computer Science
George Mason University
Fairfax, USA
kbaral4@gmu.edu

Paul Ammann
Computer Science
George Mason University
Fairfax, USA
pammann@gmu.edu

Abstract— ***Active learning is an approach that facilitates learning through collaboration, communication and discussion in the classroom. Flipped classroom is a variant of active learning where lecture materials are recorded for viewing outside the classroom thereby freeing up class time for problem solving and discussions. In this demo, we show how we use active learning in the context of teaching the testing concept of property-based testing.***

Keywords—property-based testing, active learning, flipped classroom

I. Introduction

"Active learning" is a broad term that encompasses teaching approaches where students talk, write, collaborate, move, etc. inside the classroom. This contrasts with traditional classrooms where seated students listen passively and perhaps watch slides or videos. Active learning improves student attitudes, engagement, and learning in STEM discipline [1]. One popular variant of active learning is a flipped classroom, where some lecture material is recorded for viewing outside of the classroom, hence freeing up class time for working problems, perhaps supplemented with impromptu mini-lectures as the need arises. In this demo, we show how we use active learning in the context of teaching property based testing in JUnit.

Demo Goal: Learning outcome for demo attendees is to be able to adopt an active-learning module in their own classes.

The learning outcomes for students in the JUnit module are: a) identify constraints in documentation appropriate for property-based testing and b) encode these in a suitable testing framework (e.g. JUnit Theories).

The public artifacts for this module can be accessed from an author's (Ammann) course syllabus page, which is linked from the Introduction to Software Testing website [2].

II. Aspects of the Active Learning Module

Active classroom exercises are more effective if students prepare. Hence, students should be encouraged to preview the topic before class. To facilitate this, the class schedule lists topics for each class meeting as well as assigned reading corresponding to that topic. For JUnit, Chapter 3.1-3.3 of Ammann and Offutt's Software Testing book was assigned as reading. Encouraging students to prepare with quiz questions on the reading is good practice.

The class schedule links to "ShowMe" videos that are created by an instructor (Ammann, in this case) and a student. In these videos the pair works through an exercise. Students are motivated to participate in these videos by nominal consideration at the time final grades are assigned, and students have reported anecdotally that such videos are helpful. For JUnit theories, the ShowMe video is available here. This video introduces the topic and shows the problem set-up followed by discussion to address plausible but incorrect solutions.

Students apply the material they are learning through in-class exercises - i.e. the "active learning" part. Students work in small groups. For JUnit theories, we spend a few minutes as a class brainstorming appropriate properties. We eventually visit the appropriate JavaDoc to obtain the definitive contracts. As a class, we choose one property and refine it mathematically. Then each group implements the property as a JUnit theory, ideally on a whiteboard, as that makes it far easier to track group progress. This process repeats with variations as time allows, with attention drawn to likely distractors that lead to defective theories. We focus on having students explain their solutions to us (and the class) instead of simply telling them what they should do.

Students are assigned homework to practice the materials learned in class. We encourage collaboration among students on homework because evidence shows that students learn more when they work collaboratively (peer-learning), they enjoy it more, and it matches industry practice; very few software engineers work alone. We also award the students a small bonus credit for collaboration.

Finally, to assess learning, we quiz students on the topic they learned the previous week. We post quiz solutions are posted on a discussion board (Piazza) so that students can study, question and comment.

References

[1] Freeman, S. et al., "Active learning increases student performance in science, engineering, and mathematics". Proceedings of the National Academy of Sciences, 111(23), 8410–8415, June 2014. https://dx.doi.org/10.1073/pnas.1319030111

[2] *https://cs.gmu.edu/~offutt/softwaretest/*

978-1-7281-1075-2/20/$31.00 ©2020 IEEE
DOI 10.1109/ICSTW50294.2020.00073