Pex4Fun: A Web-Based Environment for Educational Gaming via Automated Test Generation

Nikolai Tillmann, Jonathan de Halleux
Microsoft Research
One Microsoft Way
Redmond, WA, USA
Email: {nikolait,jhalleux}@microsoft.com

Tao Xie
Department of Computer Science
University of Illinois at Urbana-Champaign
Urbana, IL, USA
Email: taoxie@illinois.edu

Judith Bishop
Microsoft Research
One Microsoft Way
Redmond, WA, USA
Email: jbishop@microsoft.com

Abstract—Pex4Fun (http://www.pex4fun.com/) is a web-based educational gaming environment for teaching and learning programming and software engineering. Pex4Fun can be used to teach and learn programming and software engineering at many levels, from high school all the way through graduate courses. With Pex4Fun, a student edits code in any browser – with Intellisense – and Pex4Fun executes and analyzes it in the cloud. Pex4Fun connects teachers, curriculum authors, and students in a unique social experience, tracking and streaming progress updates in real time. In particular, Pex4Fun finds interesting and unexpected input values (with Pex, an advanced test-generation tool) that help students understand what their code is actually doing. The real fun starts with coding duels where a student writes code to implement a teacher’s secret specification (in the form of sample-solution code not visible to the student). Pex4Fun finds any discrepancies in behavior between the student’s code and the secret specification. Such discrepancies are given as feedback to the student to guide how to fix the student’s code to match the behavior of the secret specification.

This tool demonstration shows how Pex4Fun can be used in teaching and learning, such as solving coding duels, exploring course materials in feature courses, creating and teaching a course, creating and publishing coding duels, and learning advanced topics behind Pex4Fun.

I. INTRODUCTION

Teaching and learning programming and software engineering have received a lot of attention from researchers and educators. Various programming environments such as Alice [13], Scratch [12], [11], and Greenfoot [9], have been provided for instilling fun into students’ programming-learning experiences, especially for beginner learners. These programming environments have achieved significant success [22], [6] in helping teach and learn programming concepts for beginner learners. However, these environments typically target at some specialized programming languages other than mainstream programming languages. In addition, these environments primarily target at teaching and learning programming without focusing on software engineering.

As part of our research efforts on educational software engineering [24] and browser-based software for technology transfer [3], we have developed a web-based educational gaming environment for teaching and learning programming and software engineering, called Pex4Fun! [19] (denoting Pex for Fun) for mainstream programming languages such as C#, Visual Basic, and F#. It works on any web-enabled device, even a smart phone [20]. It comes with an auto-completing code editor, providing a user with instant feedback similar to the code editor in Microsoft Visual Studio. It is a cloud application with the data in the cloud, enabling a user to use it anywhere where Internet connection is available.

New learners of programming can play games there to master basic programming concepts. Learners of software engineering can play games there to master advanced programming concepts and software engineering concepts. Even experienced software engineers can play games to improve their skills while having fun.

Teachers can create virtual classrooms in the form of courses by customizing existing learning materials and games or creating new materials and games; teachers can enjoy the benefits of automated grading of game exercises assigned to students.

Behind the scene of Pex4Fun, its underlying technology is called dynamic symbolic execution [7], [14], [4], which has been realized by a white-box testing tool called Pex [17], the backbone of Pex4Fun. Pex4Fun has been gaining popularity in the community: since it was released to the public in June 2010, the number of clicks of the “Ask Pex!” button (indicating the attempts made by users to solve games at Pex4Fun) has reached over 1.3 millions (1,318,839) as of August 29, 2013.

The rest of the paper is organized as follows. Section II presents the background information on the technology and supporting tool underlying the Pex4Fun environment. Section III presents the overview of the Pex4Fun environment. Section IV presents the tool usage scenarios. Section V presents related work. Section VI discusses the potential impact of the environment to the broader community.

II. BACKGROUND: AUTOMATED TEST GENERATION

We next present the underlying technology (dynamic symbolic execution) and supporting tool (Pex) for the Pex4Fun environment.

Dynamic symbolic execution (DSE) [7], [14], [4] is a variation of symbolic execution [8], [5] and leverages runtime information from concrete executions. DSE is often conducted in iterations to systematically increase code coverage such as

---

1http://www.pex4fun.com/
block or branch coverage. In each iteration, DSE executes the program under test with a test input, which could be a default or randomly generated input in the first iteration or an input generated in one of the previous iterations. During the execution of the program under test, DSE performs symbolic execution in parallel to collect symbolic constraints on program inputs obtained from predicates in branch statements along the execution. The conjunction of all symbolic constraints along an executed path is called the path condition. Then DSE flips a branching node in the executed path to construct a new path that shares the prefix to the node with the executed path, but then deviates and takes a different branch. DSE relies on a constraint solver to (1) check whether such a flipped path is feasible; if so, (2) compute a satisfying assignment — such assignment forms a new test input whose execution will follow along the flipped path.

Pex [17] is an automatic white-box test-generation tool for .NET, based on DSE. Pex has been integrated into Microsoft Visual Studio as an add-in. Pex can generate test inputs that can be integrated with various unit testing frameworks such as NUnit2 and MSTest3. Pex was applied to test a core component of the .NET architecture, which had already been extensively tested over five years by approximately 40 testers within Microsoft. The component is the basis for other libraries, which are used by thousands of developers and millions of end users. Pex found various issues in this core component, including a serious issue. Pex was used in classroom teaching at different universities as well as various tutorials both within Microsoft (such as internal training of Microsoft developers) and outside Microsoft (such as tutorials at .NET user groups) [23].

III. OVERVIEW OF PEX4FUN

The Pex4Fun environment [19] includes coding duels as the major type of games for learning various concepts and skills in programming and software engineering. Figure 1 shows a screen snapshot of the user interface of the Pex4Fun website, which shows an example coding duel being solved by a student. Figure 2 shows the workflow of creating and playing the example coding duel.

In particular, in a coding duel, a student’s task is to implement the Puzzle method (shown on the top-right side of Figure 2 and in Figure 1) to have exactly the same behavior as another secret Puzzle method, which is never visible to the student (shown on the top-left side of Figure 2), based on feedback in the form of some selected values where the student’s current version of the Puzzle method behaves differently as well as some selected values where it behaves the same way (shown near the bottom of Figure 1 and near the right-bottom of Figure 2).

In the example coding duel, the Puzzle method is public static int Puzzle(int x). The feedback given to the student on some selected input values is displayed as a table near the bottom of the screen (in Figure 1). A table row beginning with a check mark in a green circle indicates that the corresponding test is a passing test. Formally, the return values of the secret implementation and student implementation (i.e., the Puzzle method implementation) are the same for the same test input (i.e., the Puzzle method argument value). A table row started with a red circle with a cross indicates that the corresponding test is a failing test: the return values of the secret implementation and student implementation are different for the same test input. In the table, the second column “x” indicates the test input. The third and fourth columns “your result” and “secret implementation result” indicate the return values of the student implementation and secret implementation, respectively. The last two columns “Output/Exception” and “Error Message” give more details for the failing tests.

A student can solve a simple coding duel with iterations each with the following five main steps. (1) Click an example coding duel from the Pex4Fun website; then the student can see a student implementation that does not do much. (2) Click “Ask Pex!” to see how the student implementation differs from the secret implementation [16]. (3) Compare the student implementation’s result to the secret implementation’s result. (4) Analyze the differences and change the code to match the secret implementation’s results for all input values or as many input values as the student can. (5) Click “Ask Pex!” again. Repeat this process until the student wins the coding duel (i.e.,

---

2http://www.nunit.org/
no failing tests being reported in the table by Pex) or cannot make any progress.

IV. TOOL USAGE SCENARIOS

Pex4Fun (http://www.pex4fun.com/) brings programming and software engineering with fun to a student’s web browser. Enjoying fun experiences, a student can write, compile, and run code in order to learn programming concepts, practice programming and software-engineering skills, and analyze the behavior of code interactively. In particular, this tool demonstration shows how Pex4Fun can be used to teach and learn programming and software engineering via the following aspects.

- **Solve puzzles.** The main Puzzle method used in Pex4Fun can take parameters and return values. In order to run such a puzzle method, someone must provide argument values. A student can click “Ask Pex!”, and then Pex [17], the underlying test-generation engine, automatically finds interesting argument values by analyzing the code.

- **Solve coding duels.** A coding duel is an interactive puzzle. In a coding duel, a student’s task is to implement the Puzzle method to have exactly the same behavior as another secret Puzzle method. To start with a simple coding duel, the student can click an example coding duel from the website.

- **Explore course materials in feature courses.** The current feature courses include C# for fun (for C# learners), Parameterized Unit Testing [21], [18] (for developer-testing learners [25], [23]), Code Contracts [1] in .NET (for specification learners).

- **Create and teach a course.** Pex4Fun can be used to build interesting, engaging, demanding classes on mathematics, algorithms, programming languages, software engineering, or problem solving in general. Teachers can use an integrated wiki to author classes built upon puzzles and coding duels. In particular, a teacher combines existing pages into a course. The pages might have been written by the teacher or by any other author. The teacher invites students to the course by sharing a registration link with them. A course can have multiple teachers. Any user can become a student by registering for a course through the registration link. The student can then work through the pages that are part of the course.

- **Create and publish coding duels.** A user can create and publish coding duels via five main steps. (1) Sign in, so that Pex4Fun can maintain coding duels for the user. (2) Write a specification starting from a puzzle template where the user can write the specification as a Puzzle method that takes inputs and produces an output. (3) Create the coding duel by clicking a button “Turn This Puzzle Into A Coding Duel” (appearing after the user finishes editing the visible Puzzle method text by clicking the “Publish” button).
2013. Pex4Fun has provided a number of open virtual courses including learning materials along with games used to reinforce students’ learning (http://www.pex4fun.com/Page.aspx#learn/courses). Pex4Fun was adopted as a major platform for assignments in a graduate software engineering course. A coding-duel contest (http://www.pex4fun.com/icse2011) was held at a major software engineering conference (ICSE 2011) for engaging conference attendees to solve coding duels in a dynamic social contest.

The Pex4Fun environment serves as a high-impact example to show that a sophisticated software engineering technique (e.g., automated test generation) can be successfully leveraged to underpin educational gaming and automatic grading in a web-based system that can scale to hundreds of thousands of users.

ACKNOWLEDGMENT

Tao Xie’s work is supported in part by NSF grants CCF-0845272, CCF-0915400, CNS-0958235, CNS-1160603, a NIST grant, a Microsoft Research Software Engineering Innovation Foundation Award, and NSF of China No. 61228203.

REFERENCES


