Code Hunt: Gamifying Teaching and Learning of Computer Science at Scale

Nikolai Tillmann
Jonathan de Halleux
Microsoft Research
Redmond, WA
{nikolait,jhalleux}@microsoft.com

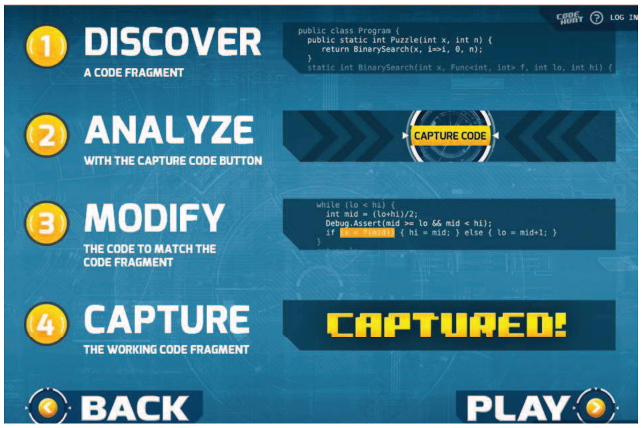
Tao Xie
University of Illinois at
Urbana-Champaign
Urbana, IL
taoxie@illinois.edu

Judith Bishop
Microsoft Research
Redmond, WA
jbishop@microsoft.com

ABSTRACT
Code Hunt (`http://www.codehunt.com/`) is an educational coding game (that runs in a browser) for teaching and learning computer science at scale. The game consists of a series of worlds and levels, which get increasingly challenging. In each level, the player has to discover a secret code fragment and write code for it. The game has sounds and a leaderboard to keep the player engaged. Code Hunt targets teachers and students from introductory to advanced programming or software engineering courses. In addition, Code Hunt can be used by seasoned developers to hone their programming skills or by companies to evaluate job candidates. At the core of the game experience is an automated program analysis and grading engine based on dynamic symbolic execution. The engine detects any behavioral differences between the player's code and the secret code fragment. The game works in any modern browser, and currently supports C# or Java programs. Code Hunt is a dramatic evolution of our earlier Pex4Fun web platform, from which we have gathered considerable experience (including over 1.4 million programs submitted by users).

INTRODUCTION
Various programming environments [1] have been provided for instilling fun into students' programming-learning experiences, especially for beginner learners. These programming environments have achieved significant success 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. Although teaching and learning basic programming concepts with specialized programming languages provide various benefits, there is a strong need of a teaching and learning platform targeting at mainstream programming languages while providing fun learning experiences. For example, such a platform shall allow students to continue the use of the platform when they move along their programming skills and experiences. In addition, such a platform shall allow students to seamlessly and smoothly transit from learning basic programming concepts and skills to learning advanced programming concepts and skills.

L@S 2014, March 4–5, 2014, Atlanta, Georgia, USA.
ACM 978-1-4503-2669-8/14/03.
http://dx.doi.org/10.1145/2556325.2567870



```
public class Program {
  public static int Puzzle(int x, int n) {
    return BinarySearch(x, i=>i, 0, n);
  }
  static int BinarySearch(int x, Func<int, int> f, int lo, int hi) {
```

```
  while (lo < hi) {
    int mid = (lo+hi)/2;
    Debug.Assert(mid >= lo && mid < hi);
    if (x < f(mid)) { hi = mid; } else { lo = mid+1; }
  }
```

Figure 1. The instructions of using Code Hunt

Code Hunt is such a publicly available platform (`http://www.codehunt.com/`). It is essentially an educational coding game (that runs in a browser) for teaching and learning computer science at scale. The game consists of a series of worlds and levels, which get increasingly challenging. In each level, the player has to discover a secret code fragment and write code for it. The game has sounds and a leaderboard to keep the player engaged. Code Hunt targets teachers and students from introductory to advanced programming or software engineering courses. In addition, Code Hunt can be used by seasoned developers to hone their programming skills or by companies to evaluate job candidates.

BRIEF DESCRIPTION OF CODE HUNT
Code Hunt is a web-based serious gaming environment (evolved from our earlier Pex4Fun [4] web platform) for teaching and learning computer science at scale. Code Hunt can be used to teach and learn computer programming at many levels, from high school all the way through graduate courses.

Figure 1 shows the screen snapshot of the instructions for using Code Hunt. The instructions can guide a new player to walk through the steps in playing the game in Code Hunt. In particular, in Step 1, from the suggested sequence of sectors in the game, the player discovers a secret code segment by selecting a sector. For example, Figure 2 shows the screen snapshot of selecting a sector. In Step 2, after the player clicks

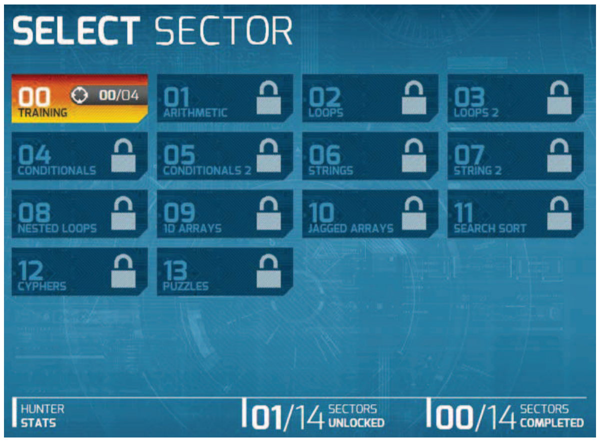


Figure 2. Selecting a sector in Code Hunt

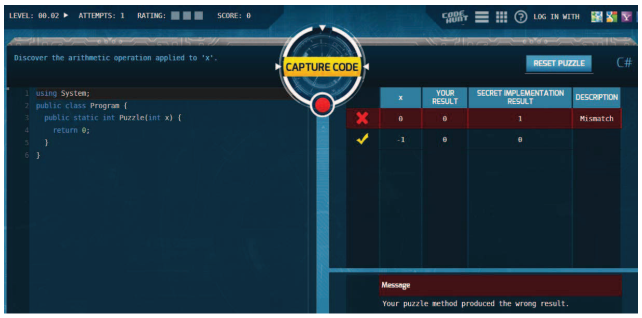
a sector, the player is presented with the player's code shown in the top-left part of the screen, as shown in Figure 3. Then the player clicks the "capture code" button shown in the up-center of the screen to analyze the behavioral differences of the secret code segment and the player's code. Then Code Hunt displays the feedback on the behavioral differences in the top-right part of the screen. In Step 3, based on the feedback, the player then modifies the player's code to match the secret code segment's behavior. Then the player iterates through Steps 2 and 3 until no behavioral differences of the secret code segment and the player's code can be found by Code Hunt. In this case, the player reaches Step 4, winning the game.

In particular, the key idea behind Code Hunt is that there is a secret code segment "under the hood" and the player is being encouraged to work towards this code segment by iteratively modifying the player's code. So close is this process to gaming, that Code Hunt is viewed by users as a game, with a byproduct of learning. Thus, new learners of programming can play games in Code Hunt to master basic programming concepts. More advanced learners of software engineering can play games to master others skills such as skills of program understanding, induction, debugging, problem solving, testing, and specification writing.

To supply feedback on behavioral differences, Code Hunt leverages an automated program analysis and grading engine based on dynamic symbolic execution [2], which has been realized by a white-box testing tool called Pex [3]. The feedback given to the player on what selected values the player's code behaves differently and the same way, respectively is displayed as a table in the top-right part of the screen (shown in Figure 3). For example, the first and second table rows in Figure 3 indicate a failing test and a passing test, respectively. The first column indicates the test input. The second and third columns "your result" and "secret implementation result" indicate the return values of the player's code and secret code segment, respectively. The last column "Exception" gives more details for the failing tests.

CONCLUSION

Code Hunt has been released only recently. However, Pex4Fun [4] (from which Code Hunt is evolved:



Discover the arithmetic operation applied to 'x'.

```
using System;
public class Program {
  public static int Puzzle(int x) {
    return 0;
  }
}
```

	x	YOUR RESULT	SECRET IMPLEMENTATION RESULT	DESCRIPTION
✗	0	0	1	Mismatch
✓	-1	0	0	

Message

Your puzzle method produced the wrong result.

Figure 3. Capturing code in Code Hunt

`http://pex4fun.com/`) was adopted as a major platform for assignments in a graduate software engineering course [5, 4]. A contest based on Pex4Fun was held at a major software engineering conference (ICSE 2011) for engaging conference attendees to solve programming problems in a dynamic social contest. Pex4Fun has been gaining high popularity in the community: since it was released to the public in June 2010, the number of clicks of the "Ask Pex!" button (similar to the "Capture Code" in Code Hunt) has reached over 1.4 million as of January 2014. Various Pex4Fun users posted their comments on the Internet to express their enthusiasm and interest to Pex4Fun.

We plan to pursue several future directions. First, Code Hunt has been built on Microsoft's cloud, Azure, and we plan to conduct load testing for Code Hunt. There is a programming contest running in China attracting 13,000+ students and we plan to leverage Code Hunt for the contest and its grading. Second, we plan to investigate whether the gamification of programming significantly improves the effectiveness and efficiency of students' learning. We have accumulated data that tracks the performance of users under Pex4Fun: within six months, we expect to collect similar data for Code Hunt and start some empirical investigations. Third, we are exploring techniques for automatically generating hints to nudge the students along when they start going off track.

REFERENCES

1. Fincher, S., Cooper, S., Kölling, M., and Maloney, J. Comparing Alice, Greenfoot & Scratch. In *Proc. SIGCSE* (2010), 192–193.

2. Godefroid, P., Klarlund, N., and Sen, K. DART: directed automated random testing. In *Proc. PLDI* (2005), 213–223.

3. Tillmann, N., and de Halleux, J. Pex – white box test generation for .NET. In *Proc. TAP* (2008), 134–153.

4. Tillmann, N., Halleux, J. D., Xie, T., Gulwani, S., and Bishop, J. Teaching and learning programming and software engineering via interactive gaming. In *Proc. ICSE, Software Engineering Education* (2013), 1117–1126.

5. Xie, T., de Halleux, J., Tillmann, N., and Schulte, W. Teaching and training developer-testing techniques and tool support. In *Proc. SPLASH, Educators' and Trainers' Symposium* (2010), 175–182.