Pop-up Java: An Augmented Reality Mobile Game to Teach Java

Richard Myers

TSYS School of Computer Science, Columbus State University, USA

Introduction

As computers become more and more ubiquitous, it becomes increasingly important and prevalent to teach coding. Coding builds important life skills such as organization, higher order thinking, self-esteem, socialization and teamwork, among many others (Hayes and Stewart, 2016). However, learning the skills necessary to design and create an efficient program from scratch takes a lot of practice that can often be confusing and frustrating to newcomers.

There are a large number of programming languages in the world, each with their own merits. The TIOBE Programming Community Index calculates the popularity of programming languages by tallying search queries on major search engines, and shows that Java is and has been the most widely used programming language for over a decade. (http://www.tiobe.com/tiobe-index/). A similar site called PYPL that aggregates results from Google Trends shows that Java makes up over 23% of programming language related queries, far outweighing any other programming languages on the list with python at 14% and PHP at 9.7% (http://pypl.github.io/PYPL.html). Popular online job board Dice.com reports that Java-related job postings are much more common than other languages (What's Hot (and Not) in Tech Skills, 2016), and the trend search on similar job board indeed.com shows that there are 2-3 or more Java-related job postings for every equivalent posting for other languages such as C++, Python, or C# (https://www.indeed.com/jobtrends/q-Java-q-python-q-c++-q-javascript-q-c%23.html).

Despite Java's popularity, pedagogical instruction of Java is typically handled in a standard lecture-and-exercises format. With the advancement of technology, there is always room for new technologies to emerge to provide new ways of learning. This paper presents an Pop-up Java, an augmented reality educational game that aims to teach Java programming constructs in an easy to use, fun, and immersive environment.

Educational Games

Educational games are games that provide more than entertainment to its users. On one hand, educational games take advantage of game design aspects that make games appealing and motivating to its users. On the other hand, educational games focus on teaching certain concept(s) or skill(s) to their users. Educational games have been shown to be effective tools in conveying ideas to students not only because it increases their motivation, but also because they can offer immediate feedback for their actions.

Games also have the capacity to not only be effective for the average student, but have been shown to be particularly effective for students who struggled with the content prior to their experience with the game (Virvou et. al, 2005). Students who find traditional instruction to be ineffective have a lot to gain from alternative approaches to instruction. Accordingly, educational games present a unique opportunity to engage learners who might otherwise be struggling with a course's content.

Augmented Reality

Augmented reality increases the immersion by displaying application information in the real world. Whereas a traditional computer game takes place in the game world "inside the screen," an augmented reality game constructs the illusion that the game is taking place in the real world, and therefore offers a more tangible experience for the user (Dunleavy et al., 2005). This is especially useful for in-game tasks that involve special reasoning, as the user is able to easily manipulate and observe the game world from different perspectives.

Motivation

This project aims to develop Pop-up Java, an educational game that leverages the power of augmented reality to make a more immersive and easily understood environment wherein students can practice Java programming concepts. Augmented reality makes the experience more immersive by showing the game objects in the real world, so the user can see, inspect, and interact with the game world from different perspectives, which helps the user to easily relate the concepts they learn to real world concepts.

In programming courses, many students struggle with understanding the basic concepts involved in object oriented programming such as defining and instantiating classes, especially if initially taught to program in a functional paradigm (Kolling, 1999).

In an augmented reality environment, instantiating a specific object in the Java program can be represented by the creation of an actual 3d object in physical space, which is easier for the aspiring programming to understand. Furthermore, since games give immediate feedback to the user, the player will be able to see any mistakes they make represented in the game world in real time and more easily understand why they happened, as opposed to a traditional programming environment. The immediate feedback offered by a game can give a student further insight into their performance, both through direct feedback such as winning or losing, or more indirect metrics such as the score on a level or the time taken to complete a challenge.

Literature Review

There are several educational games that aim to teach programming skills. Code Hunt is a browser-based game that involves writing Java code to create expected output from a function (Tillmann et al., 2014). The right side of the screen displays a set of inputs and expected outputs, and the user is required to “fix” code on the left side of the screen that almost (but not quite) gives the expected outputs. Language specific games like these are great for learning the syntax and basic functions of a language, as well as the basic computational thinking skills that go into writing a program. However, it is fairly mathematically involved compared to simpler games, and would likely only be fun to people who are already comfortable programming.

CodeSpells is an educational game developed to teach Java programming skills in a fantasy setting (Esper et al., 2013). The player controls a character capable of using magic by writing spells in Java. The game was developed by students who based the idea of the game on research they had conducted showing that, for students to learn programming outside of a classroom setting, it must be creative, exploratory, empowering, and difficult for the learners to stop once started, and something that the learners spend countless hours on. The results of testing the game with a group of 40 girls showed promising results, with the girls picking up the basic concepts involved in less than a day.

Other sites such as Codecademy combine traditional text-based instruction with “gamification” aspects that involve reward systems typically found in video games to motivate the player to progress in their studies (https://www.codecademy.com/). For example, Codecademy offers achievements for reaching certain milestones and users can earn badges to show off their progress.

Codingame is a website where a user can choose a programming language to use to fight off hordes of aliens (http://www.codingame.com). By typing in code in the editor on the right side of the screen, the player can control the ship and give it the necessary instructions to clear the level. It supports over 2 dozen programming languages and even offers dedicated forums for discussion of the game and its content.

By leveraging the systems employed by games that students are already accustomed to, the developers of the above games and websites have made their content more engaging. By offering rewards for progress, instant feedback, and easily attainable short-term goals, the student is more motivated and more likely to spend more time practicing these programming languages. However, many of these games aren't very immersive. These game worlds are completely disconnected from the real world, and the user's only interact with them through the mouse and keyboard. However, augmented reality, which bridges the gap between the real world and the game world using sensors in the user's device, has the potential to be considerably more immersive and motivating (Dunleavy et al., 2009).

Pop-up Java: An augmented reality educational game to teach Java programming

a. Game design

Pop-up Java consists of a custom-built Java interpreter being written in C# and multiple levels that address commonly taught programming concepts. The gameplay flow of a typical level is shown in Figure 1.

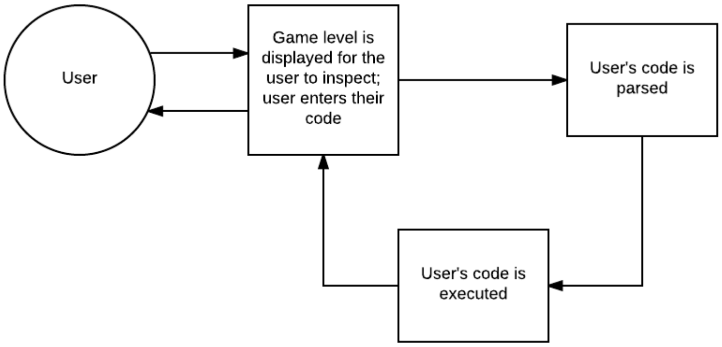
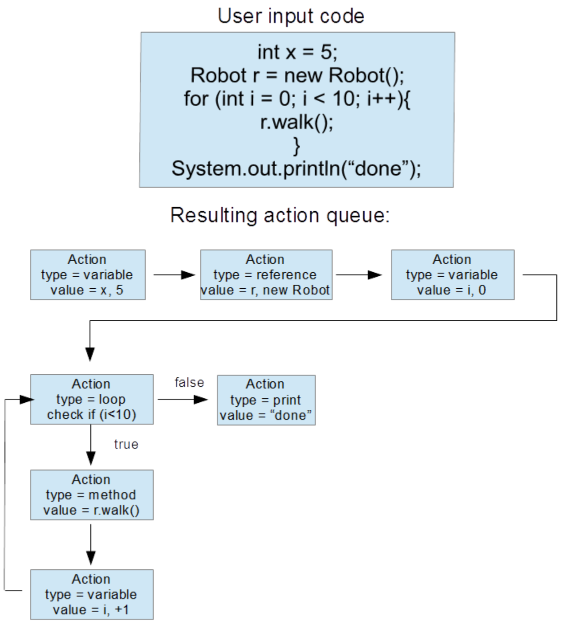

User
Game level is displayed for the user to inspect; user enters their code
User's code is parsed
User's code is executed

Figure 1: The flow of a gameplay session

The interpreter parses the Java code line by line and generates a queue of "Actions." Actions that don't influence gameplay (such as creating a variable) will have a timer to ensure the action isn't considered finished until the code has been shown onscreen long enough for the user to read the statement, and actions that do influence gameplay (such as moving an in-game object a certain distance) will be considered finished once the corresponding in-game action has finished. Figure 2 below is a representation of the action queue that would be generated from the sample code.



User input code

```
int x = 5;
Robot r = new Robot();
for (int i = 0; i < 10; i++){
r.walk();
}
System.out.println(“done”);
```

Resulting action queue:
Action type = variable value = x, 5
Action type = reference value = r, new Robot
Action type = variable value = i, 0
Action type = loop check if (i<10)
false
Action type = print value = “done”
true
Action type = method value = r.walk()
Action type = variable value = i, +1

Figure 2: A representation of a queue of actions generated from user-input code

This design is useful because it abstracts the code interpretation logic from the gameplay logic. The main game loop will still handle checking for wins and losses and executing other essential functions.

Pop-up Java allows teaching moments during which the player contributes to the background story through writing actual code. For example, one of the tasks the user is presented with early on in the game involves creating an instance of the Dog class and then creating food, water, and shelter as appropriate. The user will add the code to create a Dog object, and then follow suit for the subsequent objects that need to be created. Every time the user creates an object, a 3D representation of that object appears on screen and interact with each other as necessary (for example, the dog will eat the food when it sees it).

b. Game development

Collins (1991) lamented that in textbooks students see only worked-out solutions that do not show the false starts or dead ends that characterize real-world problem solving. Pop-up Java is an augmented reality puzzle game that provides an environment in which players can benefit greatly by “seeing” how he or she solves problems. In other words, players can see their problem solving process at work. Pop-up Java was developed in Unity, a 3D game engine, for its ease of use with Vuforia (an augmented reality software tool kit). Unity provides intuitive, easy, and most importantly, very fast way to building levels and scripting game logic. Moreover, Unity is capable of exporting the game to different platforms including Windows, Mac OS X, Linux, Android, most major VR devices, and even smart TV platforms.

Running Pop-up Java requires only mobile or laptop camera access and a pre-defined image target database that can be generated in seconds through Vuforia’s developer portal. In Pop-up Java, the player uses Java to create and manipulate in-game objects to solve these puzzles. The game world is displayed using the back-facing camera on the user’s smart device, which emulates the existence of real world objects, see Figure 3.

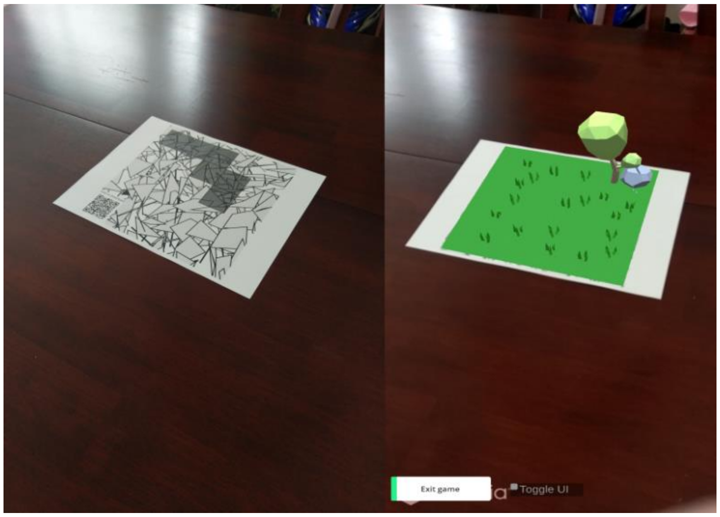
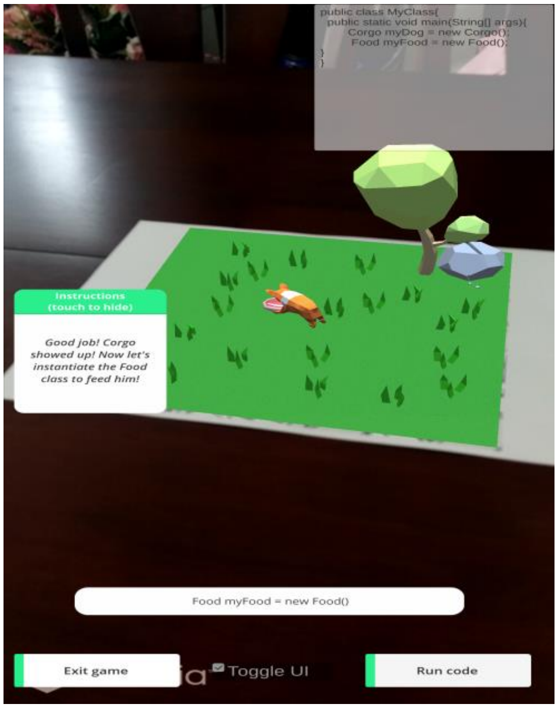


Figure 3: A composite image that shows the image target viewed through a regular camera app on the left and the image target viewed through the game on the right.

Since the game world is rendered onto the image target, the target can be manipulated (picked up or rotated) and the user can move around the image target to view the world from different perspectives. It’s also easy to understand how much more immersive a game like this is when it comes to creating objects that the user can interact with. The following image shows a screenshot of a dog eating a steak in the game, both of which were created by the user’s code. The entirety of the user’s code is shown in the top right, and the most recently executed line of code is shown at the bottom as shown in Figure 4.



```
public class MyClass{
  public static void main(String[] args){
      Corgo myDog = new Corgo();
       Food myFood = new Food();
}
}
```

Instructions
(touch to hide)

Good job! Corgo showed up! Now let's instantiate the Food class to feed him!

```
Food myFood = new Food()
```

Figure 4: Objects created by the user’s Java code appear in the game world

The use of augmented reality opens the possibility of having puzzles that must be physically manipulated in the world to understand, which offers a more immersive and motivating challenge than a typical computer game (Dunleavy et al., 2009). By giving the user the opportunity to use Java to solve spatial reasoning puzzles projected onto the real world, the programming experience becomes more tangible and easier to understand. For example, if the user has a logical error that causes the player character to move the wrong way, rather than having to debug by reading text and checking the values of variables, they'll immediately see that the player character is moving the wrong way, and understand that the currently executing line of code needs to be changed. It is worth noting that the game is played in portrait orientation on the player's device to facilitate code viewing and editing, and the user interface is togglable to ensure no obstruction of the game world itself.

Game Evaluation

Pop-up Java was tested by 20 students who are currently taking the data structures class. Results from tests have shown that most users (75% of respondents) learned something from playing the game. 90% of the respondents said that they would continue playing the game on their own, and 71.43% consider it likely that they would recommend the game to a colleague while the other 21.57% stated that they would certainly recommend the game to a colleague. Half of the respondents claimed that the game was easy to play, while the other half said they had some difficulty interacting with the game, which suggests that the user interface could be improved.

Conclusion and Future Work

Pop-up Java is an educational game in which students can experiment with Java code and grow to understand the function of the language. Pop-up Java offers immediate feedback and provides a more tangible environment for students to practice and ultimately grasp concepts. Primary results show that Pop-up java has the potential to be a very effective tool both in the classroom and at home. Future work includes developing more levels to target high level concepts in Java and add adaptation features to the game by which the game tracks the user's performance and presents the tasks based on the suer's current level of skills.

References

Collins, A. (1991). Cognitive apprenticeship and instructional technology. In L. Idol & B. F. Jones (Eds.), (pp. 121-138). Hillsdale, NJ: Lawrence Erlbaum Associates.

Dunleavy, M., Dede, C. and Mitchell, R. (20019). Affordances and Limitations of Immersive Participatory Augmented Reality Simulations for Teaching and Learning. Journal of Science Education and Technology Vol. 18 (pp. 7-22).

Esper, S., Foster, S. R., & Griswold, W. G. (2013, July). CodeSpells: embodying the metaphor of wizardry for programming. In Proceedings of the 18th ACM conference on Innovation and technology in computer science education (pp. 249-254). ACM.

Hayes, J. and Stewart, I. (2016). Comparing the Effects of Derived Relational Training and Computer Coding on Intellectual Potential in School-age Children", British Journal of Educational Psychology Vol. 86 (pp. 397-411).

Kolling, M. (1999). The Problem of Teaching Object-Oriented Programming, Journal of Object Oriented Programming Vol. 11 (pp. 8-15).

Tillmann, N., De Halleux, J., Xie, T., & Bishop, J. (2014, March). Code Hunt: Gamifying teaching and learning of computer science at scale. In Proceedings of the first ACM conference on Learning@ scale conference(pp. 221-222). ACM.

Virvou, M., Katsionis, G. and Manos, K. (2005). Combining Software Games with Education: Evaluation of its Educational Effectiveness, International Forum of Educational Technology & Society Vol. 8 (pp. 54-65).