Coding'Scool

by

Joshua Seibert, Bruce Ellicott, Sean Sutton

Submitted to the Faculty of the School of Information Technology
in Partial Fulfillment of the Requirements for
the Degree of Bachelor of Science
in Information Technology

J Joshua Seibert, Bruce Ellicott, Sean Sutton — Date 4/17/17

R Robin Carew, Faculty Advisor — Date 4/17/17

University of Cincinnati
College of
Education, Criminal Justice, and Human Services

April 2017

Table of Contents

Figures

Abstract

Ericsson and Pool (Peak: Secrets from the New Science of Expertise 2016) claim that deliberate practice, focused practiced designed by experts to work on specific drills for a skill, started at a young age is the most common factor in expert level performance. Many of those who have children and follow the trends of technology would like to give their children the basic skills they need to adopt programming during their adult lives. That is why our project: Coding'Scool aims at giving children the experience to handle logical problems and present the solutions in pseudo code. This gives children the foundation necessary to learn any coding language. The way to accomplish this is to create a game that divides essential skills into learning modules that require cumulative skills the child will gain as they progress. The most important thing is to keep the children organically engaged by implementing game theory and presenting these learning modules in fun, story related, and graphically enhanced gaming.

1. Problem Statement

1.1. Introduction

Our project team is comprised of three fathers with children ranging from 2 – 13. As our children grow, they will continually be called upon to familiarize themselves with the technology that is pervasive in our society and in our learning institutions. Each of the project team members is also a software developer challenged through school and work to learn how to code. Often this means learning the syntax of a new language but the fundamental coding concepts were our foundation for learning. We plan to provide a software-based solution to help the next generation to not only familiarize with but master the technology that will surround them.

1.2. Description

Our solution will be a game that children can play using their web browser. This game will follow a learning curriculum designed to teach users coding concepts which are applicable to multiple programming languages. To emphasize this, the application will use pseudo code as opposed to teaching in an existing language. A game provides design options which can be leveraged for maintaining user engagement and testing knowledge or learning in a reduced-stress format. The goal of this game is to prepare children for challenges they may face in their technology-dependent world. The approach our project will take in designing this game is to create a cumulative and engaging storyline and gameplay to accompany this learning.

1.3. Problem

Education is critical to preparing our children for life. Whether in personal, professional, or social contexts; the next generation will be more immersed in technology than any before it.

Intuitive design will help children to grow accustomed to picking up new technology or software and quickly learn to use it. To properly prepare our children we must also teach them how to design, create, and troubleshoot the code that runs behind the technology that surrounds them. Traditional teaching methods can and should be augmented by things like games. Games can inspire passion and pique interest in a subject that children might otherwise find complicated or boring.

1.4. User Profile

The targeted audience for our gaming application is comprised of young kids around the ages of 6-11. A secondary user of this application would be the parents/guardians of these kids. The parents will be able to set up an account for the child and link their email to it to check progress and receive notifications. The kids will be able to sign on using a user name and password, save progress in their current game; view achievements unlocked, and check their scores against other users.

Figure 1: User Profile Table

1.4.1. Potential Users:

- Child User (age 6-11)
- Parent/Guardian User

1.4.2. Software and Interface Experience:

- Child users will have general knowledge of how to interact with a web based application, and be able to perform such tasks as logging in and navigating menus.
- Parent users should have experience creating accounts using an email.

1.4.3. Experience with Similar Applications:

- Child users will be familiar with similar coding games, such as tinker and swift playground.
- Parent users will follow normal procedures for signing up with email registration

1.4.4. Task Experience:

- Child users will navigate the interface to play the main game, as well as mini games. They will also be able to update their profile, avatars, and check personal scores.
- Parent users will be able to attach an email to their child's account, after which they can check their child's progress in the game, and choose to be notified about different game events.

1.4.5. Frequency of Use:

- Child users will be able to use the app whenever they want, as long as they have an internet connection.
- Parent users will only need to set up their child's account initially, after which they can choose if they need to monitor in game progress from their email.

1.4.6. Key Interface Design Requirements that the Profile Suggests:

- Simple and intuitive UI
- Score tracking system
- Custom profile/avatar

1.5. Use Case Diagram

This diagram describes how members of the two user groups interact with the game application.

1.6. Technical Elements

1.6.1. Network

As a fully built Windows executable the game does not require a network connection to run. If OpenID or another authentication protocol is implemented in the future to connect users to online accounts then this will be modified.

1.6.2. Security

As a standalone Windows application with no connection to any networks and no stored personal information/data the player's experience during the game is insulated from malicious attacks.

1.6.3. Software

Game was built using the Unity engine, which we were able to download for free. We are able to deploy the game to many different platforms using Unity build settings.

Pre-built Assets

Unity offers a marketplace for finding game assets that are pre-built.

Below is the list of assets we imported and the function they provide:

- Ferr2D Terrain Tool for level construction
- Unity Standard Assets for Robot Kyle and miscellaneous sprites used for game objects (portals)
- Easy AI for animations and pathfinding
- Simple Drag and Drop for basic GUI functionality
- Green Forest for puzzle map creation
- Database Control for login functionality

1.7. Objectives/Deliverables

Figure 3: First Semester Deliverables

Figure 4: Second Semester Deliverables

1.8. Budget

Taking advantage of free development software and our team's unbilled time allowed us to produce this game without any cost. To demonstrate the "real world" cost we have compiled estimates of what each aspect of such a project would cost in a production environment.

Figure 5: Budget Comparison

Item	Cost for Coding'Scool	Item	Industry Average Estimates
Hosting/Server Hardware		Hosting/Server Hardware	
CECH Sandbox	$0.00	AWS	$150.00
Labor		Labor	
Design	$0.00	Design	$8,750.00
Development	$0.00	Development	$12,000.00
Licensing		Licensing	
Unity (x3)	$0.00	Unity + addons (x3)	$6,000.00
Totals	$0.00		$26,900.00

1.9. Testing

1.9.1. Overview

This section encompasses our testing procedures for our Unity game Coding'Scool. Here we discuss the scope and requirements of our testing, our procedure in which we documented each test, testing schedule, and wraps up with testing results.

1.9.2. Scope

The scope in which we tested our program is usability of Coding'Scool in the Unity platform on Windows. Individual tests are based on the various requirements needed by the application. Each test focuses on the function and success of various individual parts of the application, and also tests the usability of the application as a whole (i.e. one piece does not break another piece).

1.9.3. Requirements

Functional Requirements

- Application allows for user to create a unique username and password
- User is allowed to log in using username and password
- Overworld map allows user to interact with it and progress through the game
- Each map allows user to get to different puzzles, culminating in a final quiz that completes the level
- Each Puzzle or mini game is fully functional and does not crash or break another puzzle

Non-Functional Requirements

- User interface is easy to use and navigate
- Overall aesthetic of the application is fluid and cohesive

1.9.4. Procedure

The process for testing consisted of:

- Creating a test scenario with multiple test cases
- Creating a document write up showing what is being tested, what should be expected as a result if working correctly, and the actual results of the test
- Write up of bugs found, and steps that will be taken to correct them

1.9.5. Testing Schedule

Figure 6: Tests Scheduled

Team Member	Type of Testing	Timeline	Occurrence
Sean Sutton	Stability	Oct 2016 – April 2017	After every new major push to project.
Sean Sutton	Game functionality	Oct 2016 – April 2017	Weekly
Josh Seibert	UI	Oct 2016 – April 2017	Weekly
Josh Seibert	Portal Function	Feb 2017 – Mar 2017	Weekly
Bruce Ellicott	Game functionality	Oct 2016 – April 2017	Weekly
Bruce Ellicott	Database	Oct2016 – Jan 2017	Test until completed

1.9.6. Testing Report

Figure 7: Tests Run

Tester	Date	Test	Expected	Actual	Pass/Fail	Error
Bruce Ellicott	1/20/16	Quiz Game Launches	Quiz Game launches with no errors	Quiz game started	Pass	
Bruce Ellicott	1/20/16	Quiz Game functionality	Quiz game shows right answer	Quiz game showed correct answer	Pass	
Josh Seibert	1/17/16	Main Menu buttons	Both buttons launch correct scenes	Start launches VortexHub and login launches login page	Pass	
Josh Seibert	2/2/16	VortexHub portals, button function	All portals send player to respective world, back button send to Main Menu	All onclick methods function	Pass	
Josh	2/7/16	Collide events	Instead of click, have	Clicks	Pass	

Seibert		for VortexHub portals work	player jump into portal objects to proceed to level	deprecated. As player collides with portal, level is loaded		
Josh Seibert	2/8/16	Portals in VortexHub have labels	Each portal in the VortexHub has an easy to read label	Labels are visible but not easy to read	Fail	Large labels cause load time delays
Josh Seibert	2/9/16	Portals in VortexHub have labels	Each portal in the VortexHub has an easy to read label	Simple text labels are easy to read	Pass	
Josh Seibert	2/15/16	Each level's puzzle and quiz portals function	Each level has two objects which port the player to one of two games	All portals now work.	Pass	Some quiz portals were pointing to deprecated puzzle levels
Josh Seibert	3/21/16	Back button is on all scenes except Main Menu, Puzzles, and Quizzes	VortexHub and each Level have a back button in the upper right section or screen	Back button is there	Pass	Back button does not function in all levels
Josh Seibert	3/21/16	Back button works properly	Using the back button should send to the previous scene	Previous scene loaded (from all scenes) when back button pressed	Pass	

1.10. Conclusion

Coding'Scool aimed at reaching children in a way that will be interesting and genuinely appealing to them. This "game" teaches a child to innovate and create with technology. Children are our future and constructing software is likely to be a part of theirs. Games like ours will spark interest in the field of software development and help prepare the next generation for future careers.

References

Ericsson, Anders, and Robert Pool. *Peak: Secrets from the New Science of Expertise.* Boston, New York: Houghton Mifflin Harcourt, 2016.

Ericsson, K. Anders, Ralf Th. Krampe, and Clemens Tesch-Romer. "The Role of Deliberate Practice in the Acquisition of Expert Performance." *Psychological Review*, 1993: 363-406.

OWASP. *Open Web Application Security Project.* 2016. https://www.owasp.org/index.php/Main_Page (accessed October 2016).

Unity Technologies. *Unity.* 2016. https://unity3d.com/ (accessed October 2016).