Lab 2 - PolyMorpher Prototype Product Specification

Kevin Santos

Old Dominion University

CS411W

Professor Thomas Kennedy

26 February 2018

Version 1

Table of Contents

List of Figures

List of Tables

Lab 2 - PolyMorpher Product Specification

1. Introduction

Programming is a technical skill that requires one both retain and apply the skill in practice. It can be very intimidating for those not committed to learning it. Programming requires a logical angle that laymen in this fields are not used to thinking. Traditional forms of teaching Object Oriented Programming (OOP) and problem-solving skills at Old Dominion University (ODU) and other colleges/universities are not producing the desired results. PolyMorpher is a programming game that is designed to teach the player complex OOP concepts and problems solving skills. For PolyMorpher to succeed, it must address the deficiencies and mistakes traditional academic learning has.

1.1 Purpose

PolyMorpher will teach OOP concepts and problem solving skills in an engaging an interactive way. PolyMorpher will be a single player game that will make use of both Tangible User Interface (TUI) and a management simulator. Both of these components mimic the aspects found in modern computer-based games.

PolyMorpher will target ODU Computer Science (CS) students as its main end users. PolyMorpher will be designed and implemented by a group of students from the ODU CS department. Since this game will be designed under the jurisdiction of the ODU CS department, ODU will own and distribute it. CS students will also be targeted customers because they will be the ones to be first exposed to using PolyMorpher. The customers for PolyMorpher will also extend to instructors and individuals interested in programming within ODU and other colleges/universities.

PolyMorpher is intended to provide the necessary background and knowledge in OOP and problem solving skills to those that are motivated. PolyMorpher is not intended to supplement the entire learning content for the CS curriculum at ODU.

1.2 Scope

Programming can be an arduous skill to learn. Because of this, ODU programming students often dropout or switch majors due to the rigor required to learn OOP and problem solving skills. Traditional educational tactics towards teaching OOP concepts are not effective to new learners.. If students are not able to find the necessary help to succeed in intro level CS courses, besides failing them or switching majors, they can potentially stop attending college.

PolyMorpher will consist of three important concepts:

- Realist Approach
- Improvement on Teaching
- Balanced Gameplay

The Realist Approach allows the game to use applicable and retractable examples to teach the player OOP. Improvement on Teaching will consist on facilitating the teaching of complex OOP concepts to the player. The Balanced Gameplay concept will apply both the gaming and learning experience. These implementations will allow PolyMorpher prototype to provide the necessary tools and features to the user so that they can have a good background to be successful in intro level CS courses.

[This space intentionally left blank]

1.3 Definitions, Acronyms, and Abbreviations

API: Application Program Interface

API Book: An list of functions that the play can reference and use in their code

API Index: The menu where the user can decide rather they see the API of suggested functions for their level or all functions suggested to be used in the entire game

Assets: Game assets include everything that can go into a game, including 3D models, sprites, sound effects, music, code snippets and modules, and even complete projects that can be used by a game engine.

Coding Interface: The in game IDE where the user writes their scripts

Computer: a programmable electronic device designed to accept data, perform prescribed mathematical and logical operations at high speed, and display the results of these operations

Computer Programming: a process that leads from an original formulation of a computing problem to executable computer programs

Computer Science (CS): the science that deals with the theory and methods of processing information in digital computers, the design of computer hardware and software, and the applications of computers

Design: an outline, sketch, or plan, as of the form and structure of a work of art, an edifice, or a machine to be executed or constructed

Git: version control system for tracking changes in computer files and coordinating work on those files among multiple people

GitLab: web-based git repository manager the includes wiki and issue tracking

Gradle: an open-source build automation system that was designed for multi-project builds

GUI: Graphical User Interface

JavaScript: a programming language commonly used in web development where the the code is processed by the client's browser

Management Simulator: a way to simulate the management of a game in an organized fashion

MySQL: an open source multi-user database management system

Non-Technical Game: user-friendly gameplay able to be utilized by non-technical users

Non-Technical User: user who lacks formal education or knowledge in computer science, computer programming, object-oriented programming, or problem solving skills

Object-Oriented Programming (OOP): A schematic paradigm for computer programming in which the linear concepts of procedures and tasks are replaced by the concepts of objects and messages

ODU: Abbreviation for Old Dominion University

Platform: an integrated set of packaged and custom applications tied together with middleware

PolyMorpher: a programming game that focuses strictly on teaching OOP and problem solving skills

Problem Solving: the process of finding solutions to difficult or complex issues

Programming Game: a video game which incorporates elements of computer programming into the game, which enables the player to direct otherwise autonomous units within the game to follow commands in a domain-specific programming language

Regression Testing: a type of application testing that determines if modifications to the application have altered the application negatively

Software Development Kit (SDK): a set of software development tools that allows the creation of applications for a certain software package

Student Involvement: the amount of physical energy students exert and the amount of psychological energy they put into their college experience

Student Progression Dilemma: the problem of CS majors at ODU not advancing through the CS course schedule in order to graduate with a CS degree

TUI: Tangible User Interface

Ubuntu: open-source Linux operating system

Unity: a popular game development platform

User-Friendly: easy to comprehend by non-technical users

Virtual Machines: emulations of computer systems that provide functionalities of physical computers

Web Application: a client-server computer program in which the client (including the user interface and client-side logic) runs in a web browser

Wiki: a website on which users collaboratively modify content and structure directly from the web browser

1.4 References

Batten, C. (Narrator). (2017). CS410 Dungeon Escape Demo (Short Version) [Online video]. Online: YouTube. Retrieved from https://www.youtube.com/watch?v=ynhdd1IKgps

Batten, C. (Narrator). (2017). CS410 Project Dungeon Demo [Online video]. Online: YouTube. Retrieved from https://www.youtube.com/watch?v=ynhdd1IKgps

Batten, C. (2017, November 21). CS410 Tech Demo 2 (Download Source Code). In PolyMorpher. Retrieved from http://www.cs.odu.edu/~410silver/references.html

Batten, C. (2017, November 29). VersionControlFlow. In draw.io. Retrieved December 21, 2017, from https://www.draw.io/?state=%7B%22ids%22:%5B%221IQj6SYJqC6YLAK_qMRVIQkHiUmr9laBu%22%5D,%22action%22:%22open%22,%22userId%22:%22108692003133590583047%22%7D#G1IQj6SYJqC6YLAK_qMRVIQkHiUmr9laBu

Batten, C. (2017, October 26). CS410 Dungeon Escape Demo (Download Source Code). In PolyMorpher. Retrieved from http://www.cs.odu.edu/~410silver/references.html

Batten, C. (2017, October 26). CS410 Dungeon Escape Demo (Play Now). In PolyMorpher. Retrieved from http://www.cs.odu.edu/~410silver/references.html

Edraw. (2017, May 12). Standard Flowchart Symbols and Their Usage. In Edraw Visualization Solutions. Retrieved from https://www.edrawsoft.com/flowchart-symbols.php

Everitt, C. (2017, September 6). Current Process Flow. In draw.io. Retrieved December 21, 2017, from https://www.draw.io/?state=%7B%22ids%22:%5B%220B-5KdQEdqLUPdnBFUnp2V05uMEE%22%5D,%22action%22:%22open%22,%22userId%22:%22108692003133590583047%22%7D#G0B-5KdQEdqLUPdnBFUnp2V05uMEE

Everitt, C., & Dang, D. (2017, September 24). currentProcessFlow. In draw.io. Retrieved December 21, 2017, from https://www.draw.io/?state=%7B%22ids%22:%5B%220B3Bc95zBWXg9TFZ6X0FMU1NTdEk%22%5D,%22action%22:%22open%22,%22userId%22:%221086920031335905830 47%22%7D#G0B3Bc95zBWXg9TFZ6X0FMU1NTdEk

Everitt, C., Santos, K. & DeArce, N. (2017, November 27). Work Breakdown Structure (WBS). In draw.io. Retrieved December 21, 2017, from https://www.draw.io/?state=%7B%22ids%22:%5B%220B-5KdQEdqLUPWnNoSHhIUGg2OTQ%22%5D,%22action%22:%22open%22,%22userId%22:%221086920031335905830 47%22%7D#G0B-5KdQEdqLUPWnNoSHhIUGg2OTQ

Everitt, C., Santos, K. & DeArce, N. (2017, October 13). ProcessFlowDiagram_silver. In draw.io. Retrieved December 21, 2017, from https://www.draw.io/?state=%7B%22ids%22:%5B%220B_xBnZ1ge4PlZTVjV3h6Y2pGSWc%22%5D,%22action%22:%22open%22,%22userId%22:%221086920031335905830 47%22%7D#G0B_xBnZ1ge4PlZTVjV3h6Y2pGSWc

Few, S. (2008, February 5). Practical Rules for Using Color in Charts. In Perceptual Edge. Retrieved from http://www.perceptualedge.com/articles/visual_business_intelligence/Rules_for_using_color.pdf

Kennedy, T. (2017, September 6). kennedyData. In Google Drive. Retrieved from https://drive.google.com/drive/u/1/folders/0B_xCQd8Vk2BnSU1hNnJwSXB1NEE

O'Neill, M. (2017, March 6). Computer Science Before College. In Computer Science Online. Retrieved from https://www.computerscienceonline.org/cs-programs-before-college/

Riley, P. (2017, September 14). Using Games to Introduce Programming to Students [PowerPoint slides]. Retrieved from http://www.cs.odu.edu/~410silver/references.html

Stokes, J. (Narrator). (2017). CS410 Programming Game Pitch [Online video]. Online: YouTube. Retrieved from https://www.youtube.com/watch?v=QBvgzFgZaOQ&feature=youtu.be

Stokes, J. (2017, October 9). CS410 Programming Game Pitch (Download Source Code). In PolyMorpher. Retrieved from http://www.cs.odu.edu/~410silver/references.html

"The Benefits of Video Games." abcnews (2011, December 26). Retrieved October 19, 2017, from http://abcnews.go.com/blogs/technology/2011/12/the-benefits-of-video-games/ Good-Morning-America

Santos, K., Riley, P. & Dang, D.(2017. December 7) Risk matrix and description tables in Design Presentation. Retrieved from https://docs.google.com/presentation/d/1oY9lkSAHvg2OIRkljYJNZWCqVTbiw45STKglsJUQjJI/edit#slide=id.g283e74317a_0_177

Team Silver. (2017, December 13). Prototype PowerPoint Presentation. In *PolyMorpher*. Retrieved from https://docs.google.com/presentation/d/e/2PACX-1vSidnjCKAuVEtKshHkyO7A-OfW3qWIKRkxcp0em412WwL1ig6SFmnqrMUyHr8-FMvzvaRjmcKYiCytq/pub?start=false&loop=false&delayms=3000&slide=id.g25ab9a9d23_0_1542

Team Silver. (2017, November 21). Design PowerPoint Presentation. In *PolyMorpher*. Retrieved from https://docs.google.com/presentation/d/e/2PACX-1vSllslBDmSvRfMI9nbrp0RmRaPRsHNz7YWDfKNiF5sg15cp7ycQ774MuMgm4G4qhR6hohTiUQrrjRdo/pub?start=false&loop=false&delayms=3000&slide=id.g25ab9a9d23_0_1542

Team Silver. (2017, October 25). Feasibility PowerPoint Presentation. In *PolyMorpher*. Retrieved from https://docs.google.com/presentation/d/e/2PACX-1vReG6Sodx-

gVFro1ByYMOYHSyiSRiU5HW-Su-PyMVGO8F4CQ7pY49tB_pJecVApruksoGaP_00ozhmR/pub?start=false&loop=false&delayms=3000&slide=id.g25ab9a9d23_0_1542

Unity Technologies. (2017, August 10). Company Facts. In Unity. Retrieved from https://unity3d.com/public-relations

Unity. (2016, July 6). Unity - Scripting API. In Unity. Retrieved December 21, 2017, from https://docs.unity3d.com/530/Documentation/ScriptReference/index.html

Unity. (2017, October 11). Asset Store. In Unity. Retrieved December 21, 2017, from https://www.assetstore.unity3d.com/en/

12 Free Games to Learn Programming. (2016, April 25). In Mybridge. Retrieved from https://medium.mybridge.co/12-free-resources-learn-to-code-while-playing-games-f7333043de11

1.5 Overview

The purpose of this product description paper is to provide the general description of the prototype for PolyMorpher. This paper will describe PolyMorpher's architecture and functional capabilities. It will also provide detailed descriptions of its five external interface aspects. Section 3 will address the product specification requirements submitted in a separate document.

2 General Description

The main objective of the PolyMorpher prototype is to provide a working demonstration of the final product. The way to accomplish this is by implementing the components and features required to build the full product. The final product will have the same capabilities as the prototype in that it will have the same gameplay style as other programming games. It will also

teach problem solving skills and OOP concepts like abstraction, polymorphism, inheritance, and encapsulation.

2.1 Prototype Architecture Description

The architecture found in the PolyMorpher prototype will be the same as the one in the final product. The PolyMorpher prototype will be separated into three components: PolyMorpher website, PolyMorpher application, and the Unity file structure

- PolyMorpher website: This website will contain the PolyMorpher game application that users can download as an executable file. It will also provide a play guide for the users to inform themselves of the rules and tips for the game.
- PolyMorpher application: The application will be a downloadable executable file that will allow the user to play it on their local machine. The application can be downloaded from the PolyMorpher website
- The Unity file structure: PolyMorpher prototype will contain the entire unity file structure. It includes the “StreamingAssets” directory which will be accessed and modified when the user uses the “morphing” option in the game objects.

Figure 1 will illustrate the components of the architecture structure for the PolyMorpher prototype and their interactions.

[This space intentionally left blank]

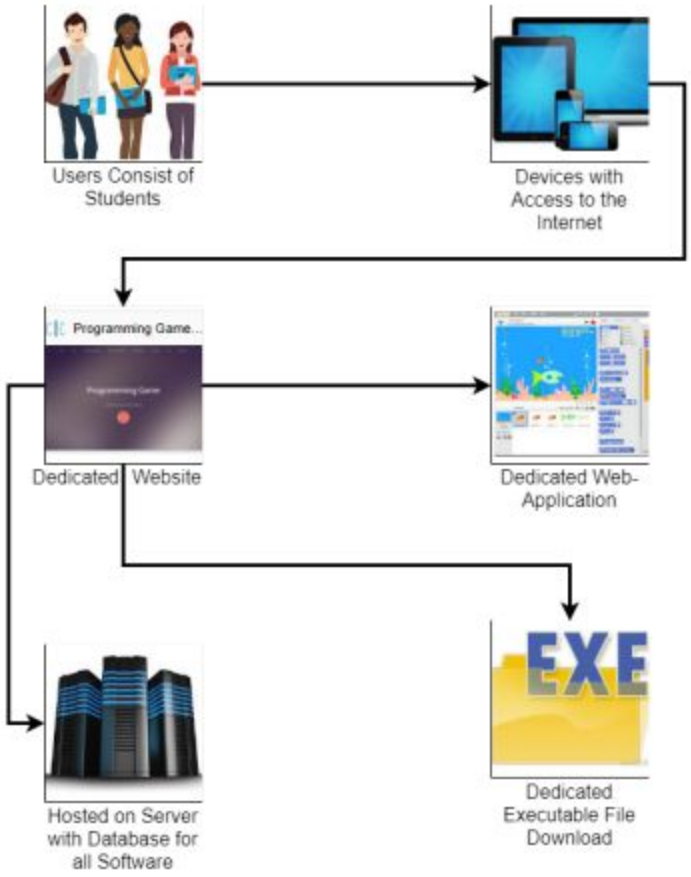

Figure 1: Major Functional Component Diagram

2.2 Prototype Functional Description

Both the PolyMorpher prototype and final product will be able to be downloaded from its website. The prototype version will be compatible with most common operating systems: Windows, Linux, and MacOS. Table 1 summarizes the 16 possible differences between the PolyMorpher Prototype and the final product.

[This space intentionally left blank]

Elements	Description	Real World Product	Prototype
Teaches Polymorphism	Provision of a single interface to entities of different types		
Teaches Abstraction	Technique for arranging complexity of systems		
Teaches Encapsulation	Building of data with the methods that operate on that data		
Teaches Inheritance	When an object or class is based on another object or class, using the same implementation		
Single Language Taught	A single programming language will be focused on C#.		
Single Player	Focused on an experience targeted to interact with only one player		
Downloadable .EXE File	Desktop application version of the game		
Game Assets	Primary components that are used as building block to construct the more complex features and levels of the game		
Developed Story	Narrative used to drive progression or direct player throughout a more guided/linear experience		
Portable Compiler	Code compiler used to run player-made code on the fly in game		
Tutorial Section	Precursor series of levels meant to help the player adjust to the in-game toolset given to them and also prep them with knowledge of the language(s) they will be working with		
Multiple Platforms	Version support for multiple operating systems (Windows, Mac OS, Linux)		
Sandbox Level	Open level where the player has access to all tools at once and can build their own level sequences and puzzles		
Player-Made Content	Variant of Sandbox Level, potentially allows the player to share custom levels with one another		
Multiple Player	An experience geared toward multiple players interacting with a game environment together		
Web Application	Web based version of the game running in-browser		
Multiple Languages Taught	Alternative programming languages for the player to use and learn in-game		

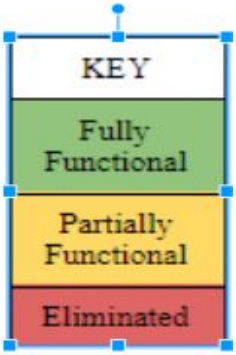

Table 1: Features of the Completed Product against the Prototype

It will be guaranteed that the prototype will be able to teach some of the most important OOP concepts as stated at the beginning of section 2. In addition, the player will be able to benefit from a story progression and a tutorial section so that they can see their amelioration as they move from one level to another while they play through the game. One of the partially functional features that might be implemented on the prototype is a sandbox level. This is where the player is allowed to have access to all the tools available in the game so that they can design

their game levels and puzzles. Lastly, the eliminated features of PolyMorpher for both the prototype and the final product are multiplayer, multiple languages taught, and the web application.

2.3 External Interfaces

PolyMorpher will require the implementation of five external interfaces: Hardware Interface, Software Interface, User Interface, API Book Interface, and Compiler Interface.

2.3.1 Hardware Interface

PolyMorpher will be implemented as a 2-Dimensional game that will draw minimal computing resources. Polymorpher is designed to operate on the user's PC and will be optimized to perform with Intel's 4th generation i3 processor.

2.3.2 Software Interface

PolyMorpher will run as a desktop application. It will be compatible with Windows, Linux, and MacOS operating systems as stated in section 2.2. PolyMorpher will be cross-platform because of the ability Unity has to build applications in different and various platforms.

2.3.3 User Interface

The user interface will be comprised of three main components: Computer screen, Keyboard, and a Mouse.

- Computer screen: A computer screen is required to display the game. The game display will be under the Unity gameplay interface. Figure 2 will illustrate a technical gameplay demonstration of the prototype screen interface.

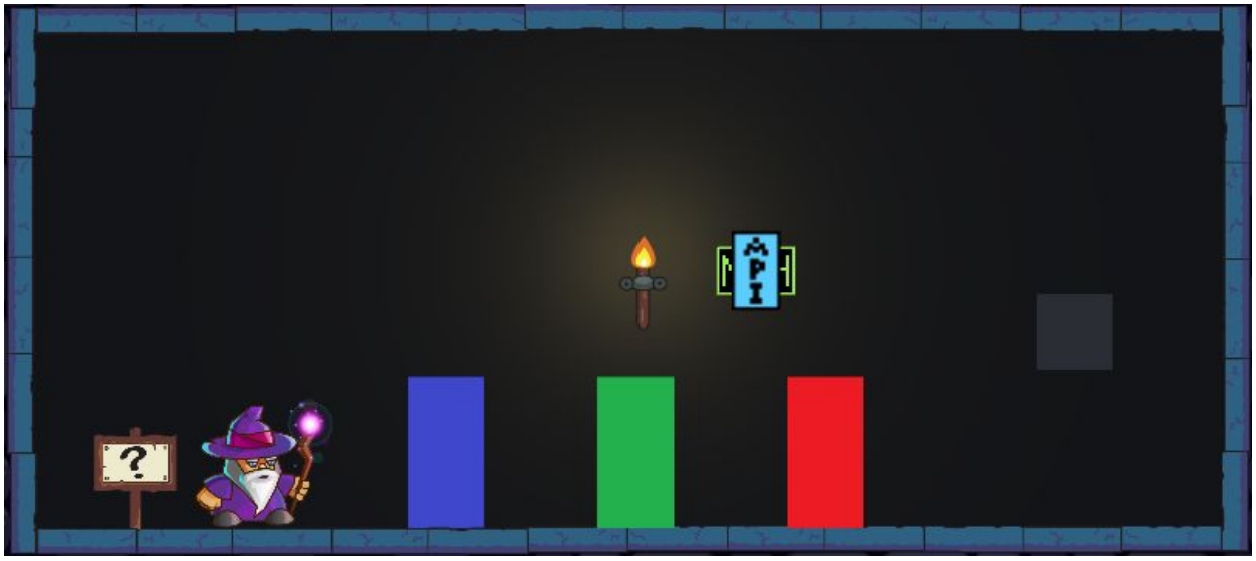

Figure 2: Technical Gameplay Demonstration for the Computer Screen

- Keyboard: The keyboard will be used to move the character and approaching objects to "morph". It will also be used to type the required syntax for C#. The keyboard will also be needed to type the text in the text editor box for every object the user decides to "morph".
- Mouse: The mouse will be needed to navigate through the TUI of the game. It will also be used to select the object to "morph" as well as accessing the API Book interface.

2.3.4 API Book Interface

The API Book interface provides a variety of tools to help the user complete the puzzles and challenges in each level. The API Book Algorithm illustrates the functionalities for the PolyMorpher's API Book option in the game. This algorithm consists of three simple procedures. First, the player selects the API icon which will be an image of a book that appears in a textbox. Then, they select a function inside the provided API Book. Lastly, the function the

player selects appears in the textbox which it can be editable and manipulated. Figure 3 shows the API Book Algorithm diagram.

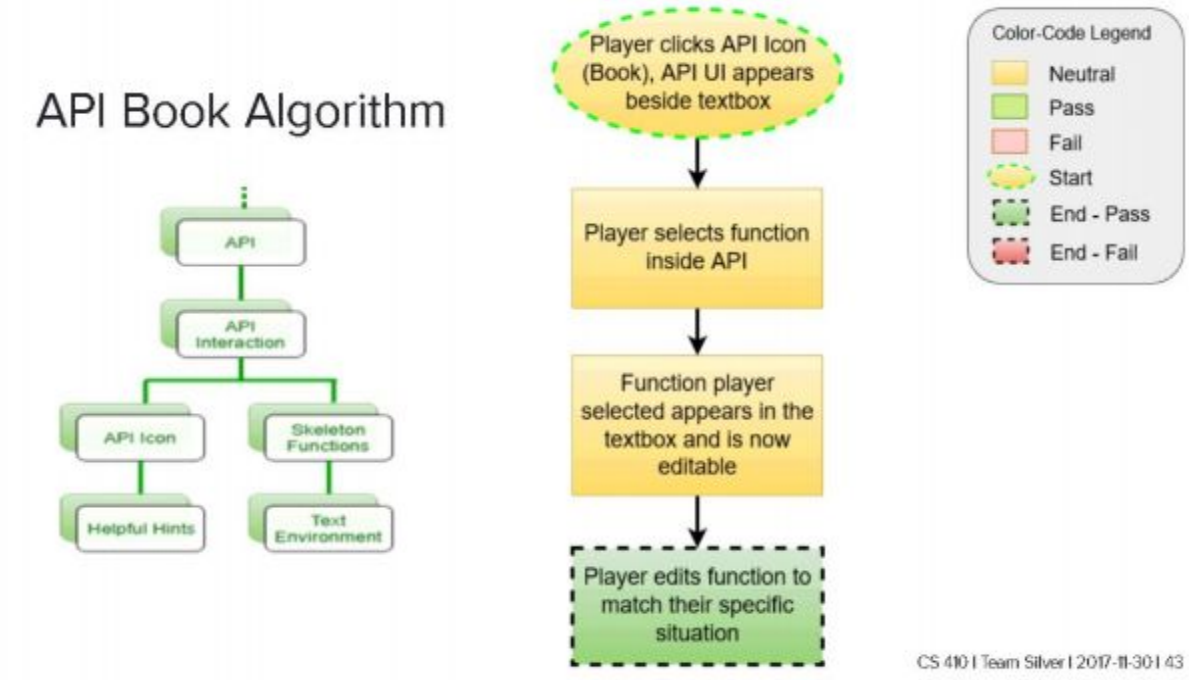

Figure 3: Dataflow Diagram for the API Book Algorithm

2.3.5 Compiler Interface

The Compiler Interface is composed of a text box that contains incomplete C# code in which the user is to edit it in order for it to become syntactically correct. It becomes available when the user selects an editable object to “morph”. The compiler will check the entered code to make sure it complies with the edited object’s needed functionality. If it does, the user moves on to the next level and if not; they have to repeat the “morph” process again. The Compiler Interface contains three buttons. Figure 3 will illustrate a technical gameplay demonstration of the Compiler Interface.

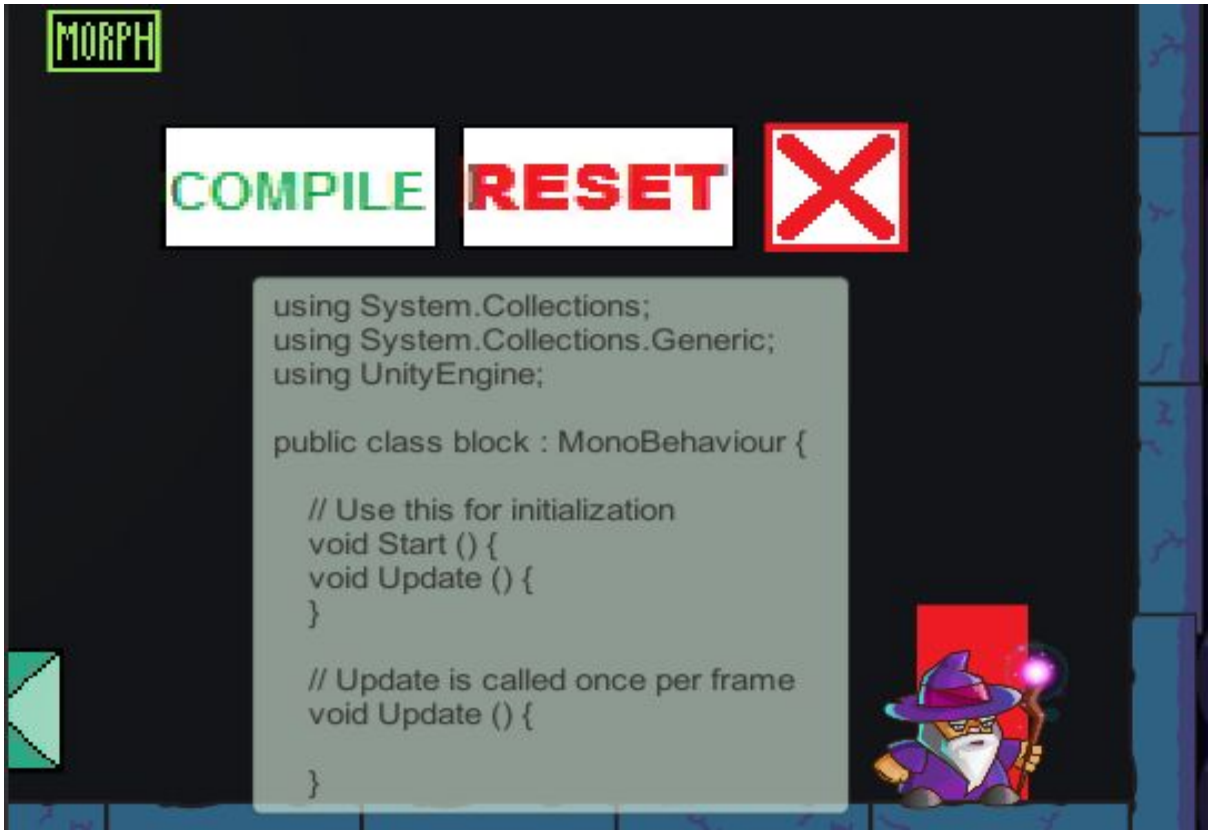

Figure 4: Technical Gameplay Demonstration for the Compiler Interface

The compiler algorithm comes into action when the user selects the compile button. It will act as a portable assembler to process the code the player enters as they "morph" their selected editable object. Figure 4 illustrates the entire overview of the Compiler Algorithm.

[This space intentionally left blank]

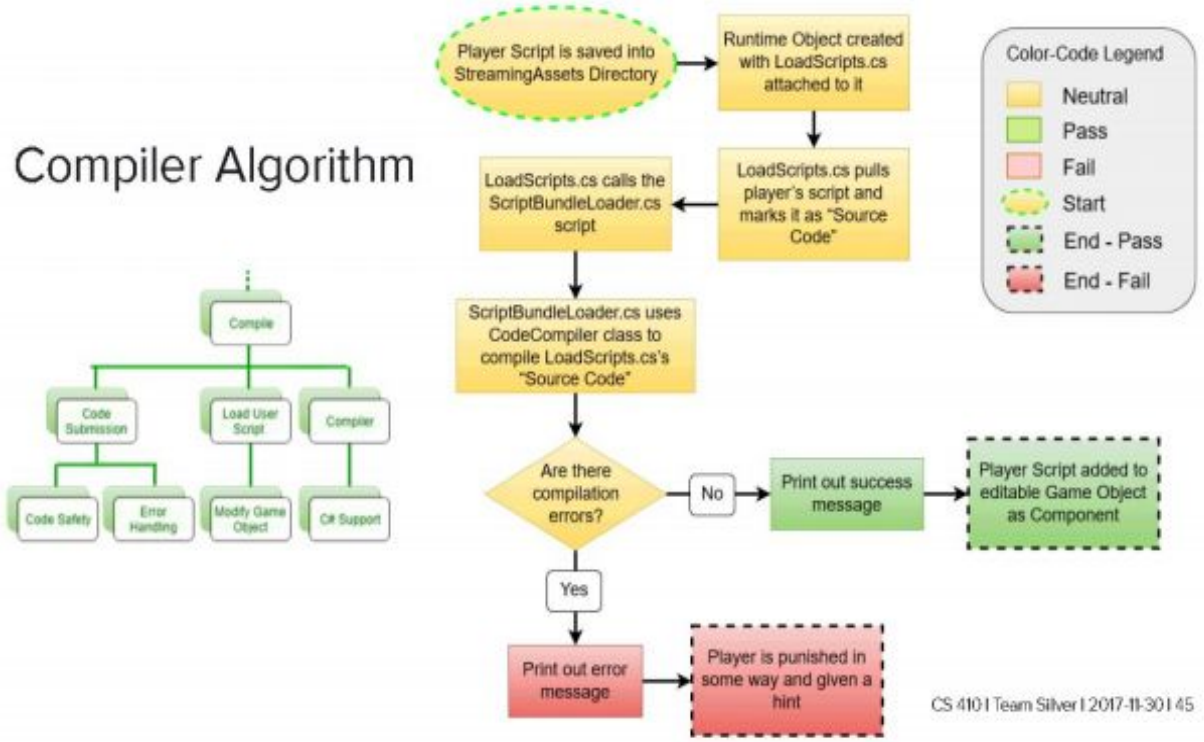

Figure 5: Dataflow Diagram for the Compiler Algorithm