



Lab IV - PolyMorpher Prototype User Manual

Team Silver

Old Dominion University

CS411W

Professor Thomas Kennedy

17 April 2018

Version 1

Table of Contents

Introduction	3
Starting the Game	4
Game Instructions	7
PolyMorpher Guide	14
Troubleshooting	18
Glossary	19

Lab 4 - PolyMorpher Prototype User Manual

I. Introduction**a. Purpose**

Students, as well as those who are generally interested in software development and programming, will enter a negative cycle of repetition and failure if they struggle to grasp the core aspects of Object-Oriented Programming (OOP). The purpose of Polymorpher is to help not only teach, but also reinforce, those core concepts and the tenants of software development as a whole. This is achieved through a series of concept-focused levels that will let the player learn and grow through free experimentation while working towards a specified goal. Each goal requires the player to use a specific tenant of OOP design in their code in order to be able to progress to the next level and eventually complete the game. It is through this interactive model, as well as the open gameplay of Polymorpher, that the player will be able to escape the negative cycle of progression.

b. Layout:

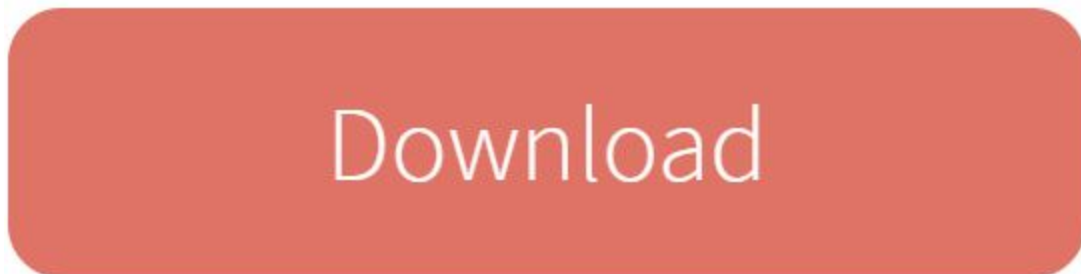
This document will break down the components of Polymorpher, as well as provide general information related to the game and services involving the game. The sections that follow will cover the process of acquiring Polymorpher, as well as the steps involved in beginning the game and ensuring its ability to run properly. Beyond that, documentation will be provided on overall controls, the core layout and structure of the game and its mechanics, and other aspects related to player interaction alongside the software's overall functionality. Also included is a section detailing a play guide to

describe the core focus of individual levels, as well as provide hints and instructions involved in the solution of said levels. Finally, there is a section providing detailed instructions on troubleshooting and software maintenance. This is followed by the Glossary section, which provides in depth descriptions of the terminology used throughout this document.

II. Starting the Game

a. Downloading PolyMorpher

The user is required to navigate to the PolyMorpher website. The user is to click on the Download tab in the upper right hand corner. There will be a large red rectangular button where the user has to click in order for the downloading process to begin.



Once the download button is pressed, the user will be prompted to answer a questionnaire regarding their name and if they agree with the End User License Agreement.

* required field.

Full Name:

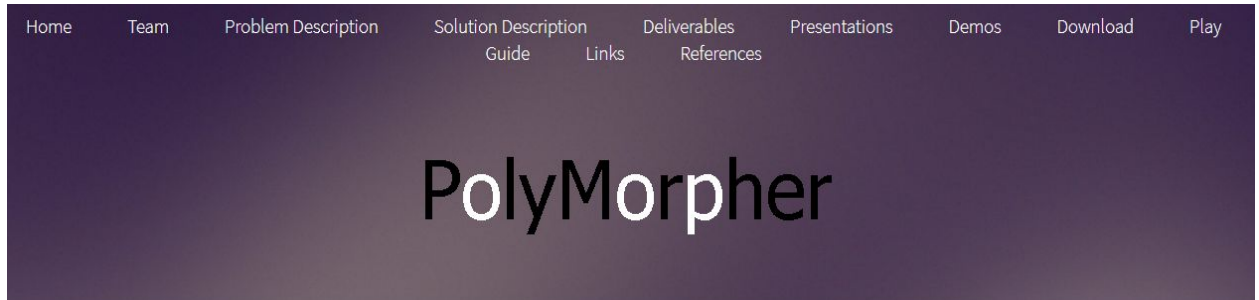
E-mail:

* Email is required

Gender: Female Male * Gender is required

PolyMorpher EULA

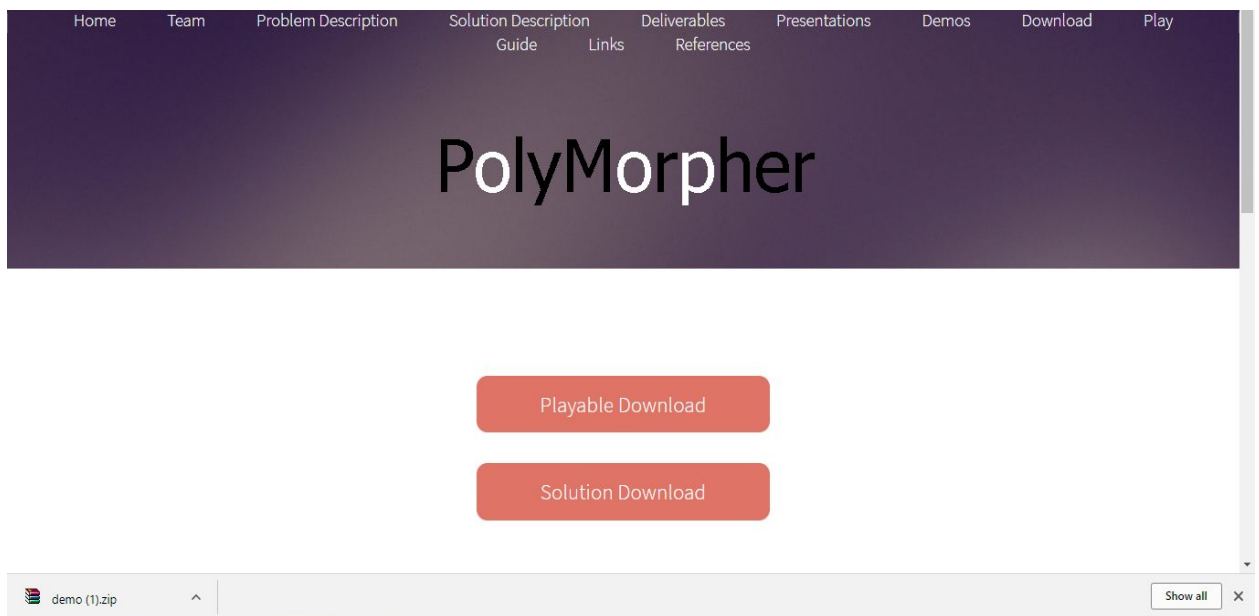
Once the user has filled out the questionnaire and agreed to the EULA terms, the user must click to download the playable version of the game.



Playable Download

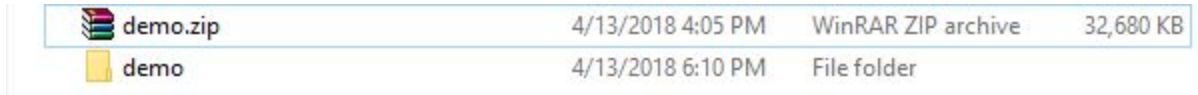
Solution Download



Once the Playable Download button is pressed, a zipped folder that contains the executable file will be downloaded to start the game.



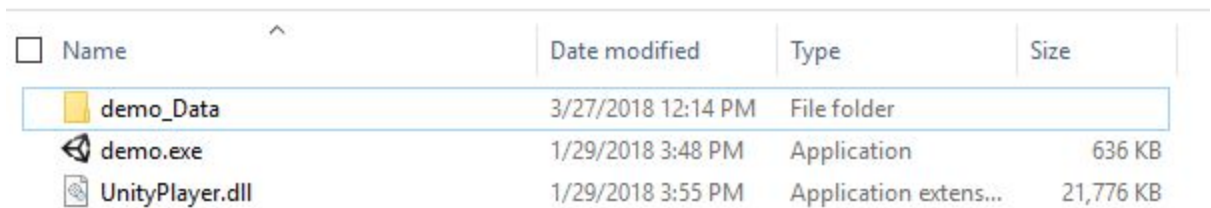
b. Start PolyMorpher




Once the demo zipped folder has been downloaded, the player will extract it.



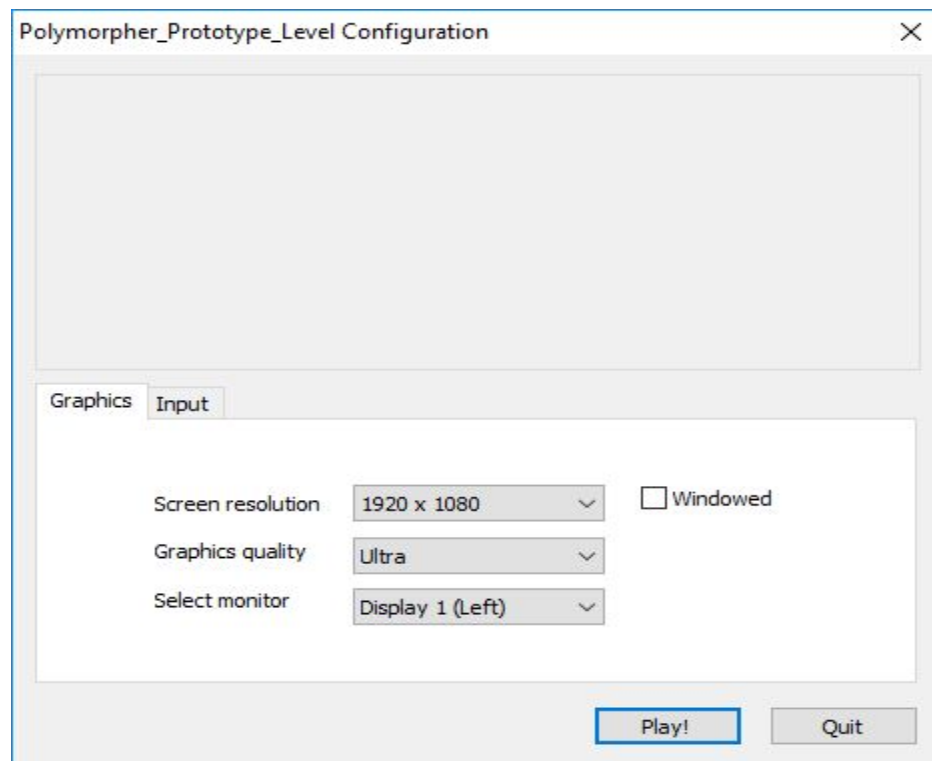
 demo.zip	4/13/2018 4:05 PM	WinRAR ZIP archive	32,680 KB
 demo	4/13/2018 6:10 PM	File folder	

Once the folder is extracted, the user will double click on the “demo.exe” file to initiate the game.



<input type="checkbox"/> Name	Date modified	Type	Size
 demo_Data	3/27/2018 12:14 PM	File folder	
 demo.exe	1/29/2018 3:48 PM	Application	636 KB
 UnityPlayer.dll	1/29/2018 3:55 PM	Application extens...	21,776 KB

When the “demo.exe” file is executed, the game will load and request for the user’s input configuration.



Once the configurations have been set at the user's desire, the user will click on the play button.



Once the game executes, the user will start the tutorial level to help them understand how the game works. The box above will instruct the user on how to begin playing the game.

III. Game Instructions

a. Introduction

Step 1. Open PolyMorpher

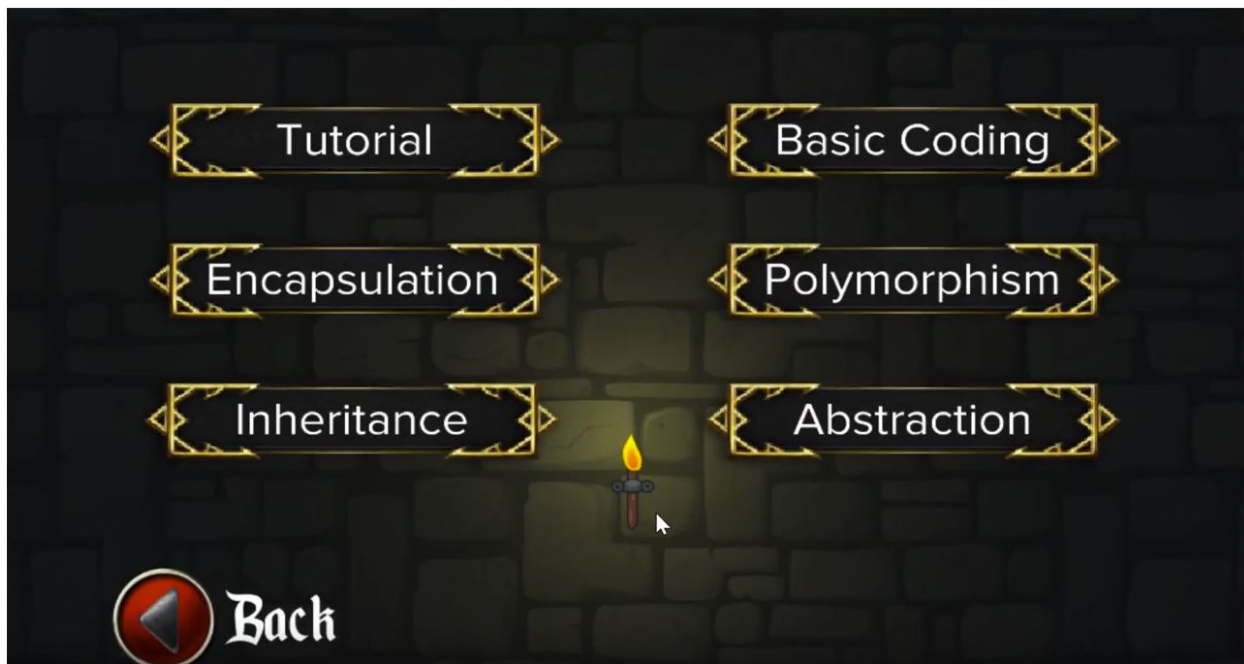
Upon opening PolyMorpher, the main title window of the game will present the three different options of Settings, Play, and Exit. Here the player can choose from the three different options, depending on what he feels like doing. The Settings button will allow the player to adjust in-game settings such as volume level, save status, load status, and various help menus for each level. The Play

button will allow the player to initially begin the game; the menu will display the different level options available. The Exit button will allow the player to leave the game by closing out the executable/application file.



Step 2. Choose Desired Level

The level menu displays all of the different levels that a player could click on in order to play each level. The different levels include Tutorial, Basic Coding, Encapsulation, Polymorphism, Inheritance, and Abstraction. The levels are split up to each cover a certain major topic that is a part of Object-Oriented Programming (OOP). Each level is described in detail pertaining to what lessons are covered in section b steps 2 and 3 of this section, and also in section IV. PolyMorpher Guide. The Back button will allow the player to go back to the main title window of the game.



b. How to Play the Game (Beginners)

Step 1. Go through the Tutorial Mechanics Level

The game will begin at the Tutorial Mechanics Level. The purpose of this level is to introduce players to the mechanics of the game. This level will teach players how to move their character, interact with selectable objects, morph objects, and use the API book. The player will be first introduced to how to move their character from the in-game dialogue window that pops up. The player can move their character with the keyboard arrows. The player is then told that they are trapped in a dungeon and that they must find a way out. The only way to escape the dungeon is to solve programming challenges. Next, the player will be taught about the selectable and read-only objects within the game, and then instructed on the goal of this level. Selectable objects are marked with a green glowing light

when the mouse is over them. These are objects that the player can edit.

Read-only objects are very similar, except they glow red and are not editable.

Next, the player will click on the block, and be presented with the in-game

Integrated Development Environment (IDE). This IDE will allow the player to

write code that affects the object they have selected. They can click “Compile” to

run their code, or if they wish to start with a blank slate they can click “Reset”.

The player will then be shown the API Book and how to use it. The API Book

will be very useful as players progress through each level. The API Book will

show the player suggested functions that can be used to complete the level. It will

also list all the APIs available within the entire game. It is possible that a function

from another level may be useful to a player regardless of what level they are

currently on.

Step 2. Go through the Tutorial Syntax Level

After the tutorial mechanics level, players will be introduced to the tutorial syntax

level. This level will teach players the basic mechanics needed to master the

game. Players will be tasked with a series of intuitive mini problems that they

need to complete to pass on to the next level. This level will introduce a series of

three switches and three boxes. The objective of this level is to have the players

match the corresponding boxes to the appropriate switches, denoted by color. The

boxes will have properties which include a transform, a rigidbody, and a box

collider. The goal is for the player to use the appropriate scripts to match the

components to move the boxes to their respective switches and open the door to

the next level. This will show the players the basic core gameplay of the game in its simplest form. This will also give players an introduction to the API book and the syntax required to manipulate the game objects throughout the levels.

c. How to Master the Game (Experts)

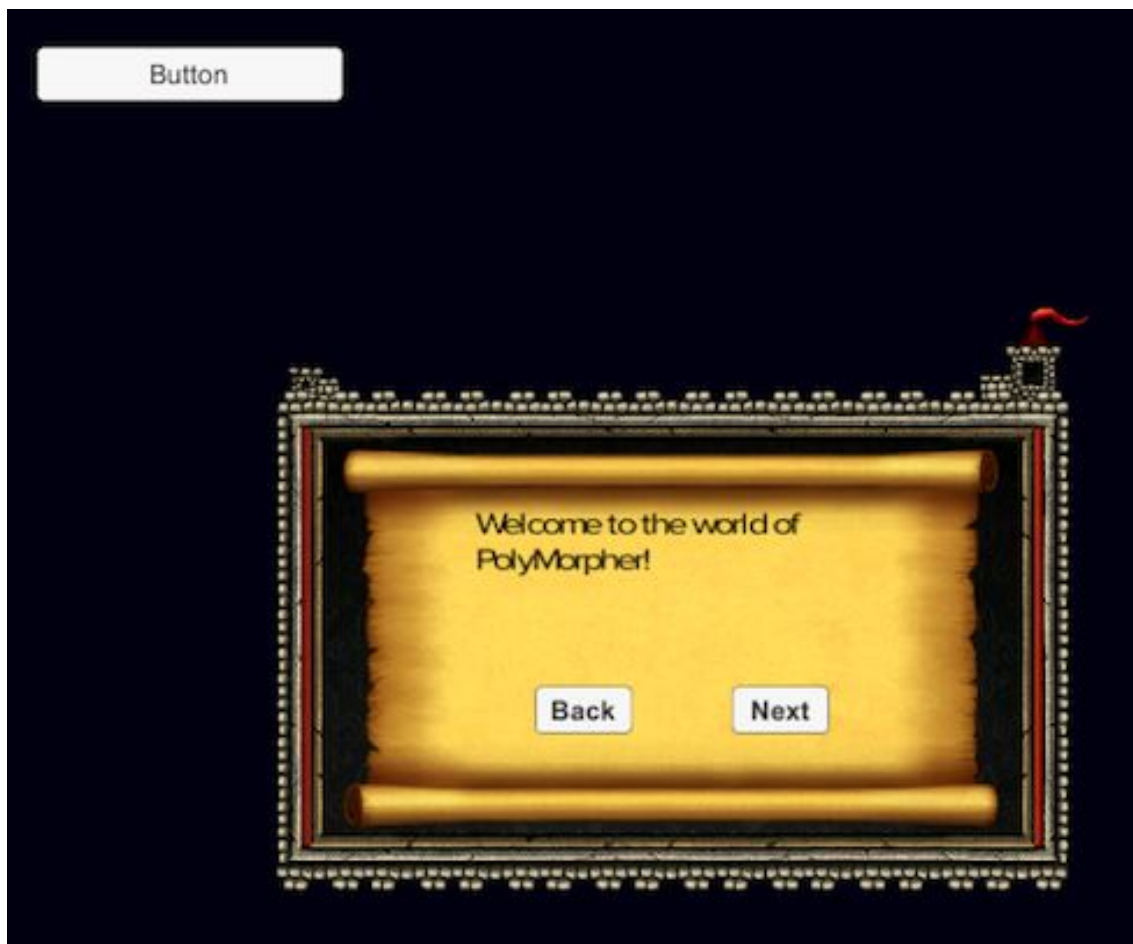
There are many ways to master the game, or become an expert player.

PolyMorpher is normally played by progressing through the levels by morphing only one editable object in each room of each level. The game can be mastered, or played at an expert experience level, by discovering new ways to beat each level rather than just following the given tips to get past each level. The player can find a way to morph different editable objects, or morph the recommended editable objects in a different way than given in the API Book for each level. Soon the expert player will be writing code to be included in the API Book.



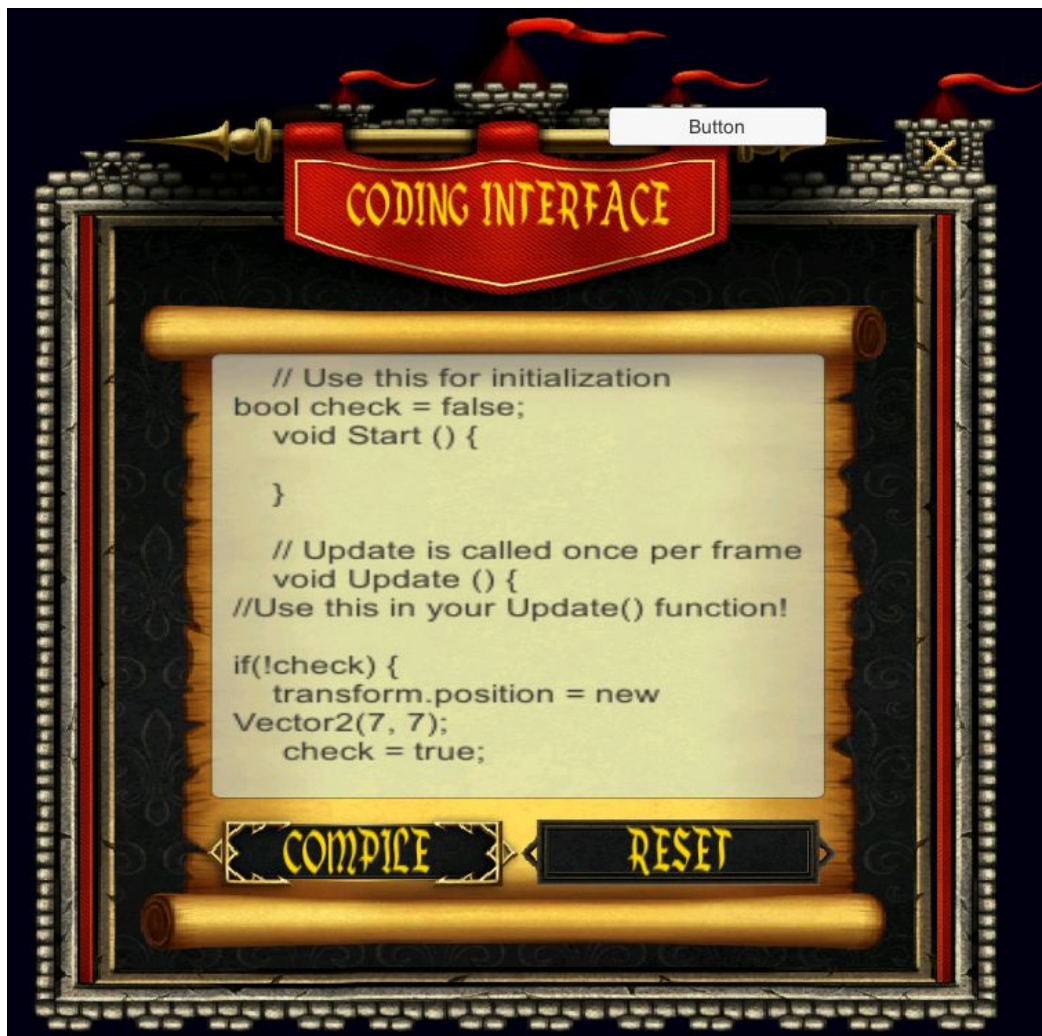
d. How to Replay a Level

Any level can be restarted, or replayed, by simply clicking the button currently labeled Button (Button will be changed to say Reset Level or Restart Level in the final prototype version). Clicking this button will reset everything in the level to its initial state and move the main character's position to where it was when the level was first started. However, the custom scripts written by the players will not be reset unless the reset button is pressed in the Coding Interface area when an editable object is clicked.



e. How to Save a Custom Script

Any custom script written by the player is automatically saved when typed inside the code editor when an editable object is clicked on. When the Compile button inside the Coding Interface is clicked, the player's code is compiled and he is notified of whether his code compiles or not. Only if the player clicks on the Reset button inside the Coding Interface will his custom code is deleted/erased; in this case, he has to start from scratch.



f. How to Exit the Level or Game

To exit the game, the user must first be at the main menu of the game. Once at the main menu, the option of exiting the game should be chosen and the game will close.

**IV. PolyMorpher Guide****a. Abstraction**

The principle of abstraction is one of four Object-Oriented Programming principles that will be taught in the world of PolyMorpher. Through the process of abstraction, a programmer hides all but the relevant data about an object in order to reduce complexity and increase efficiency. There are different levels of abstraction in object oriented programming ranging from the least abstract data types such as integers, floats, and booleans to extremely abstract data types that seem more like objects the player would see in real life such as coffeeMachine, WorkScheduler, or calculator. The

more abstract data types contain less abstract data types inside of them, allowing programmers to reuse data types to increase efficiency.

In the abstraction level of PolyMorpher, the player will be prevented from entering the next room by an orc. The player must put the orc to sleep by gathering three different plants, preparing each of them through different scripts for a potion, and then dropping them into the cauldron, which is protected by arrows. The player will have to create another script that will be used in each of the player's unique plant scripts in order to stop the arrows, put the orc to sleep, and move one step closer to escaping the dungeon.

b. Inheritance

The purpose of the inheritance level is to give the player an understanding of how objects inherit functions and attributes from existing objects. This level will have an object derived from a Vehicle class. The Vehicle class will contain all the elements needed to complete the level, which will consist of an obstacle course.

In order to complete the level, the player will need to edit the given object's code to utilize the functions and attributes it inherits from its parent class. These attributes will include wheels, an engine, and other vehicle parts. The player will choose which parts are suitable for passing a section of the obstacle course.

The course is divided into three parts with checkpoints in between. After each checkpoint, the player will be given access to a new set of vehicle attributes to pass the next obstacle. If the player chooses, they can reset the level to the last checkpoint or reset the level entirely.

c. Polymorphism

The Polymorphism level will teach the player the concept of polymorphism, one of the three pillars of object-oriented programming. Polymorphism is the ability to handle objects differently depending on their data type or class. The player will need to utilize this concept to reach the end of the level.

The player will find boxes of two types, metal and wooden. These boxes are derived from an abstract base class, Box, which has abstract methods. There are flames and ice, which the player will use to interact with the boxes. This level also has a state that renders it impossible; at this point, the level must be reset.

d. Encapsulation

The Encapsulation level teaches the user the basic principles of keeping functions and variables private to prevent unintentional interference from outside code. The level centers around a system of keys and locked doors in progressively more complex conditions. As the level progresses, monsters will be able to peer into the code of the key the player is carrying, and if the information inside is not properly kept private, the monster will steal the key from the player.

The level centers around a theme of peering, ominous eyes. The eyes symbolize the Encapsulation concepts taught in the level, “watching” the player’s code that is inappropriately made public. The level’s monsters are large, floating eyeballs with wings which have a line of vision in front of them. If the player enters their line of vision, the monster will run a key checking function to see if it has access to any of its unlocking functions. If it does, the monster will make the key vanish, forcing the player go back and

try again. The level ends once the player reaches the end of three locked corridor puzzles to a stairwell leading to the next level.

There are no checkpoints within the level. Thus, there is no way for the player to die, and the level itself is fairly short. However, there are rare cases of the player somehow glitching out of bounds due to chaotic code. Also, all keys in the level have regeneration pedestals, so if the key is taken by a monster or lost out of bounds in the level, the pedestal will create a new, identical key object in its place.

[This Space Left Intentionally Blank]

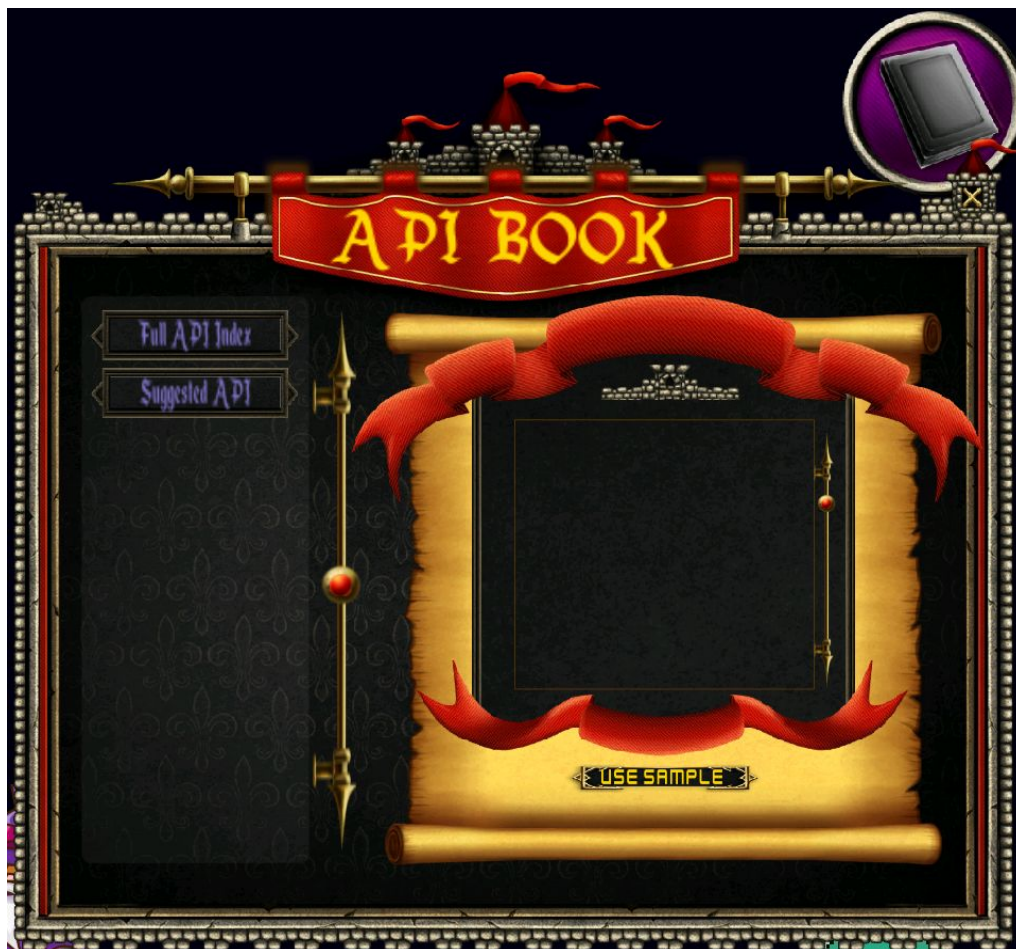
V. Troubleshooting**a. Redownload**

If unable to download the game in the first try, the user should check their internet connection and make sure they are getting a good compatible bandwidth. If the internet connection is good, the user should follow the instructions of downloading and starting the game in section II of this manual. The game should download again with no issues.

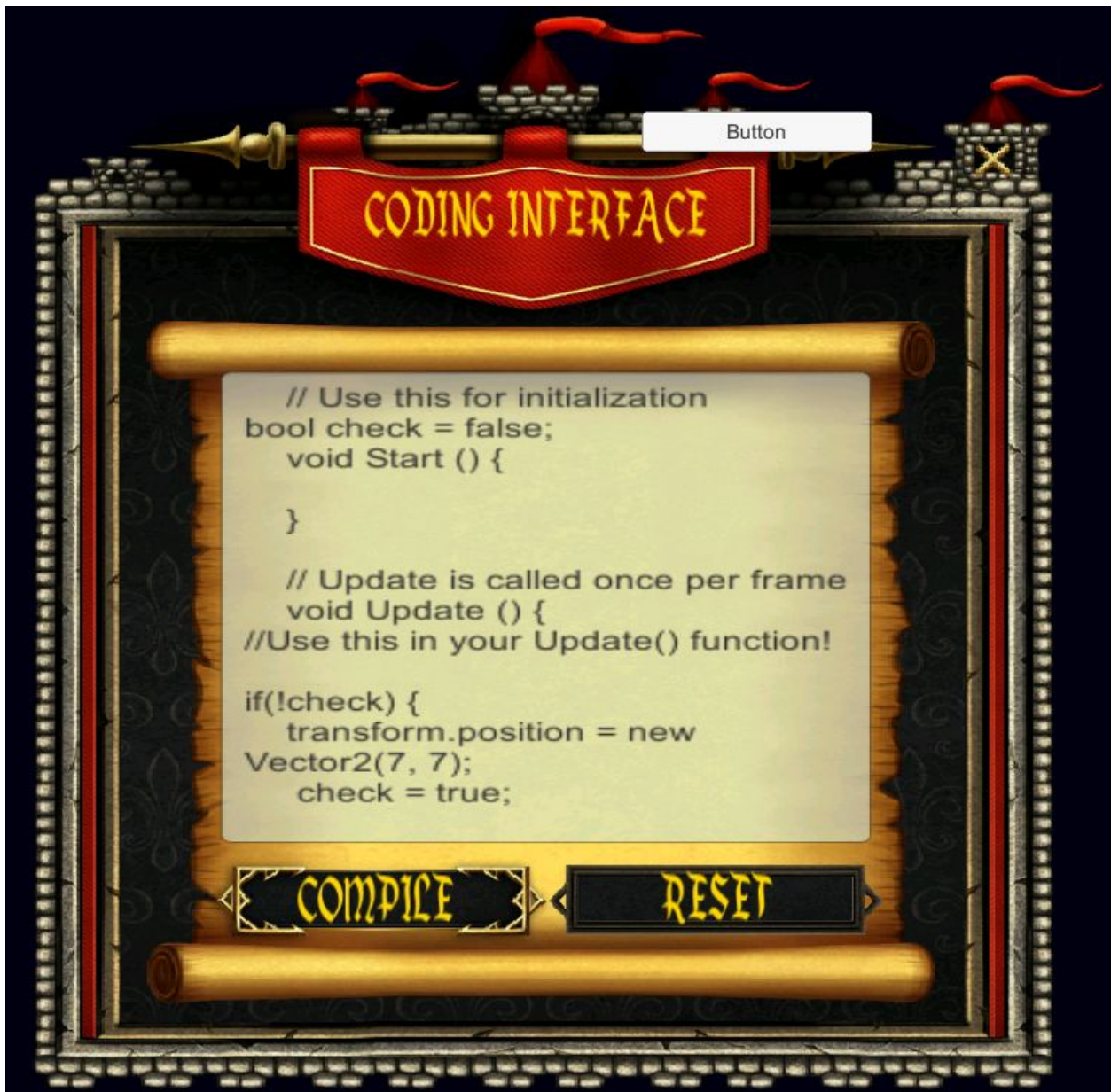
[This Space Left Intentionally Blank]

VI. Glossary

- a. **API:** Application Program Interface - A tool for assisting developers in creating applications
- b. **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
- c. **PolyMorpher:** a programming game that focuses strictly on teaching OOP and problem solving skills
- d. **API Book:** A list of functions that the player can reference and use in their code



- f. **Coding Interface:** The in-game IDE where the user writes their scripts



- g. **Problem Solving:** the process of finding solutions to difficult or complex issues
- h. **Programming Game:** a video game which incorporates elements of computer programming into the game, which enables the player to direct autonomous units.