BuzzWord

Software Design Description

Author: Tejas Prasad
Professaur Inc.
November, 2016
Version 1.0

Abstract: This document details the software design for BuzzWord, a generalized educational word game of “Boggle”.

1 Introduction

This is the Software Design Description (SDD) for the BuzzWord educational game application. This document format is based on the IEEE Standard 1016-2009 recommendation for software design.

1.1 Purpose

This document will serve as the blueprint for constructing the BuzzWord application. The design will use UML class diagrams to provide complete detail with regards to all packages, classes, instance variables, class variables, and method signatures required to create the application. UML Sequence diagrams will also be used to display object interactions of the application or in other terms, responses to user interactions or timed events.

1.2 Scope

BuzzWord will be one of the first word-games to be developed to capture the educational games market. Since additional games will be created, design decisions and tools must be constructed to develop more word games. Due to this, a framework will be created called the ***Java Framework for Language Applications and Games*** (JFLAG) which will be the foundation for other word games. JFLAG will be a slight modification of the current framework, JFXFramework, so that JFLAG can be used for future word games.

1.3 Definitions, Acronyms, and Abbreviations

1. Educational games - games that are designed to help people to learn about certain subjects, expand concepts, reinforce development, understand an historical event or culture, or assist them in learning a skill as they play.
2. Word games - games that involve making, guessing, or selecting words.

3. IEEE (Institute of Electrical and Electronics Engineers) - Pronounced “I triple E”, is the world’s largest association of technical professionals. Its objectives are educational and technical advancement of electrical and electronic engineering, telecommunications, computer engineering, and their allied disciplines.
4. Framework - An abstraction in which software providing generic functionality can be selectively used by additional user-written code, thus providing application-specific software. In case of Java, a framework typically takes the form of a collection of classes and interfaces that can play such a foundational role.
5. Graphical User Interface (GUI) - A type of user interface that allows users to interact with electronic devices through graphical icons and visual indicators such as secondary notation, instead of text-based user interfaces, typed command labels or text navigation.
6. Unified Modeling Language (UML) - A standard set of document formats for designing software graphically.
7. Class Diagram - A UML document format that describes classes graphically. Specifically, it describes their instance variables, method headers, and relationships to other classes.
8. Java - A high-level programming language that uses a virtual machine layer between the Java application and the hardware to provide program portability.
9. Sequence Diagram - A UML document format that specifies how object methods interact with one another.
10. Java Framework for Language Applications and Games (JFLAG) – A software framework developed alongside the BuzzWord game to serve as a foundation for future word games.

11. Java FX Framework (JFXFramework) – A framework created by Professor McKenna and Professor Banerjee to serve as a foundation for a JavaFX application.

1.4 References

IEEE Std 830™-1998 (R2009) - IEEE Standard for Information Technology – Systems Design – Software Design Descriptions

BuzzWord SRS – Professaur Incorporated's Software Requirements Specification for the BuzzWord educational word game application.

1.5 Overview

This Software Design Description provides constructible design for the BuzzWord application as detailed in the BuzzWord Software Requirements Specification. Section 2 of this document details the Package-Level Viewpoint which will specify the packages and frameworks to be designed. Section 3 will detail the Class-Level Viewpoint utilizing UML Class Diagrams to display how the classes should be constructed. Section 4 will detail the Method-Level System Viewpoint to describe how the methods will interact with each other. Section 5 details the file structures and formats. Section 6 will have a Table of Contents, an Index, and References. All UML Diagrams in this document were created with the VioletUML editor.

2 Package-Level Design Viewpoint

The design of this application will include the BuzzWord word game application and the JFLAG which will be used to develop the application. JFLAG will be slightly modified from the JFXFramework, notably by changing the layout given in the *AppGUI* class.

2.1 Buzz Word Overview

Figure 2.1 and 2.2 display the classes to be developed or have been developed and which packages they will reside in for the BuzzWord application and JFLAG. JFLAG includes the Jackson JSON library which was part of the JFXFramework.

Figure 2.1: BuzzWord Design Packages Overview

Figure 2.2: JFLAG Design Package Overview

2.2 Java API Usage

The BuzzWord game will be developed using the Java API. The classes to be used are listed in figure 2.3 and their descriptions are in section 2.3.

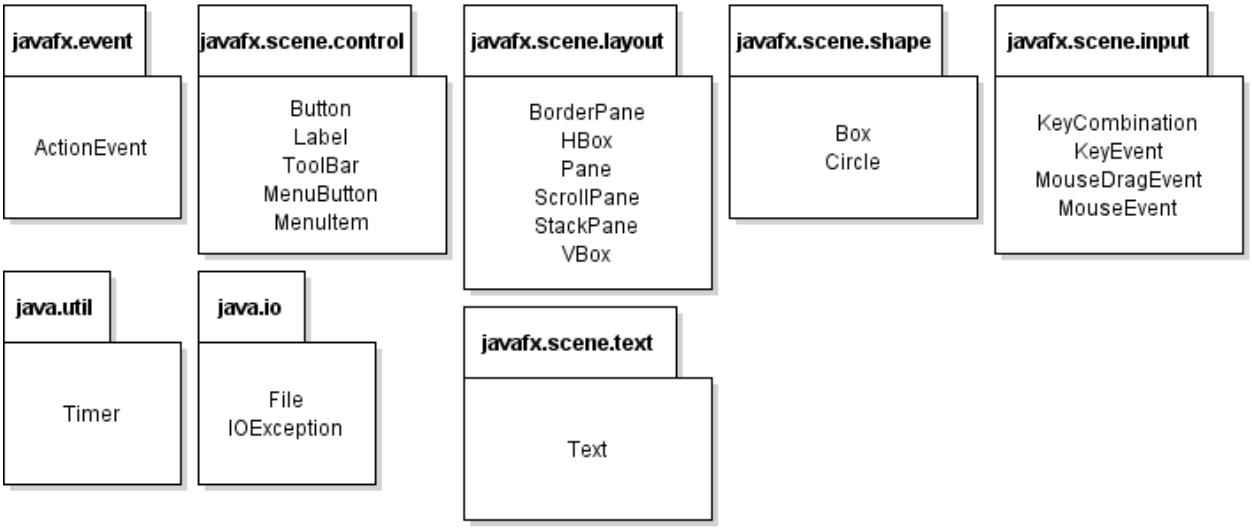

Figure 2.3: Java API Classes and Packages

2.3 Java API Usage Descriptions

Class/Interface	Use
ActionEvent	For receiving information about an event such as a button press.

Table 2.1: Uses for classes in the Java API's javafx.event package

Class/Interface	Use
Button	For different options to be used
Label	For the title of the game
MenuButton	For selecting the game mode
MenuItem	For listing all the game modes available

Table 2.2: Uses for classes in the Java API's javafx.scene.control package

Class/Interface	Use
BorderPane	For organizing the layout of the game into sections.
HBox	For organizing elements horizontally such as the mode levels.
Pane	For a container to place the game elements.
ScrollPane	For organizing more elements than the screen can display and to scroll to view the unseen elements such as in the word list and the help screen.
StackPane	For placing elements on top of another to create new elements primarily for each letter in a circle.
VBox	For organizing elements vertically such as the buttons on the right side of the application.

Table 2.3: Uses for classes in the Java API's javafx.scene.layout package

Class/Interface	Use
Box	To draw a box such as for the timer and the current score.
Circle	To draw a circle for each letter in the game.

Table 2.4: Uses for classes in the Java API's javafx.scene.shape package

Class/Interface	Use
KeyCombination	For specifying a key combination such as shift+a.
KeyEvent	For getting information about which key was pressed.
MouseDragEvent	For getting information about which letters the mouse was dragged over.
MouseEvent	For getting information about which letter was clicked.

Table 2.5: Uses for classes in the Java API's javafx.scene.input package

Class/Interface	Use
Timer	To record and countdown the time for each level.

Table 2.6: Uses for classes in the Java API's java.util package

Class/Interface	Use
File	To specify which file to use to save and load from.
IOException	For recording an error in finding or loading a file.

Table 2.7: Uses for classes in the Java API's java.io package

Class/Interface	Use
Text	To display characters or messages.

Table 2.8: Uses for classes in the Java API's javafx.scene.text package

3 Class-Level Design Viewpoint

The following UML Class Diagrams display the BuzzWord game and how it will be designed.

The UML Diagrams go from overview diagrams to detailed ones for each package.

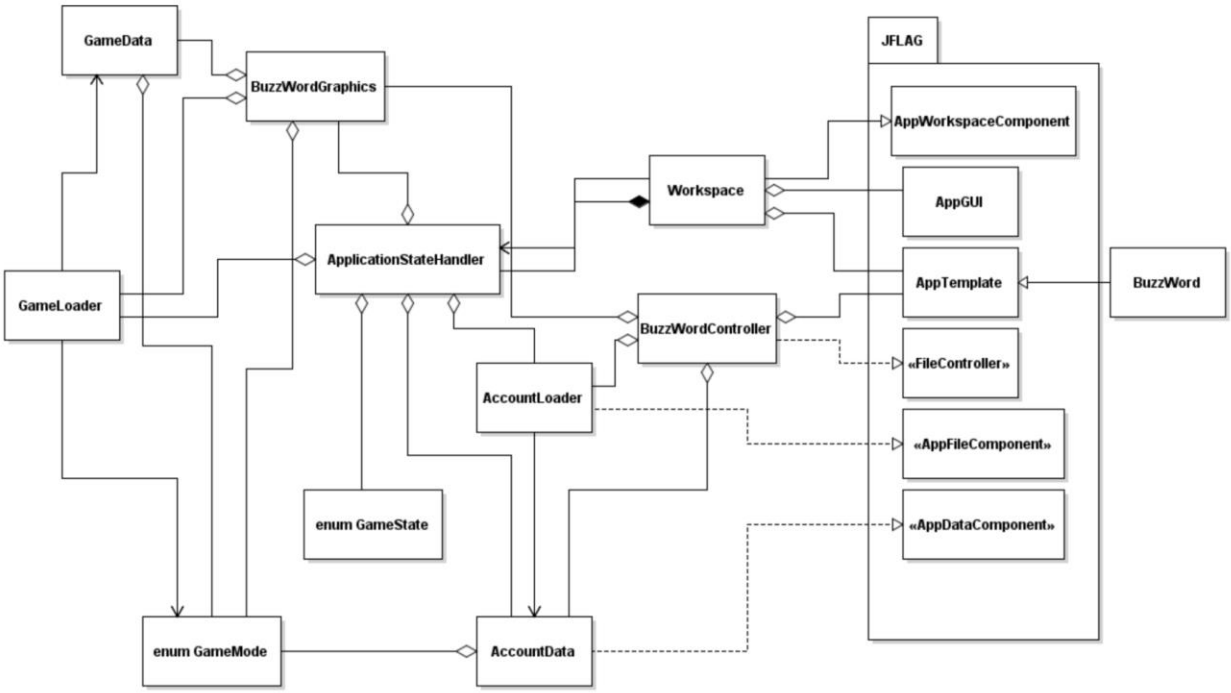

Figure 3.1: BuzzWord Overview UML Class Diagram

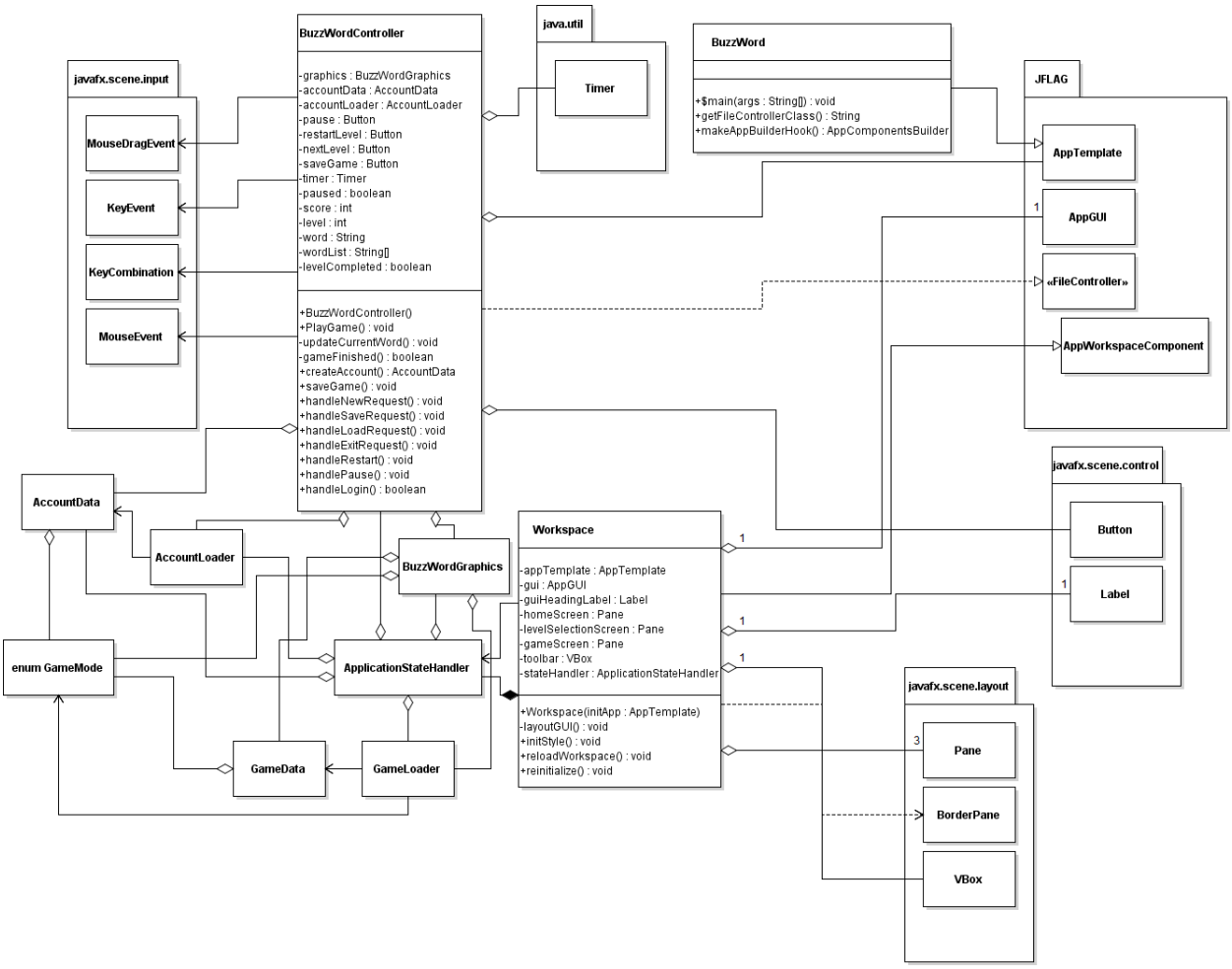

Figure 3.2: Detailed BuzzWord, BuzzWordController, and Workspace UML Class Diagrams

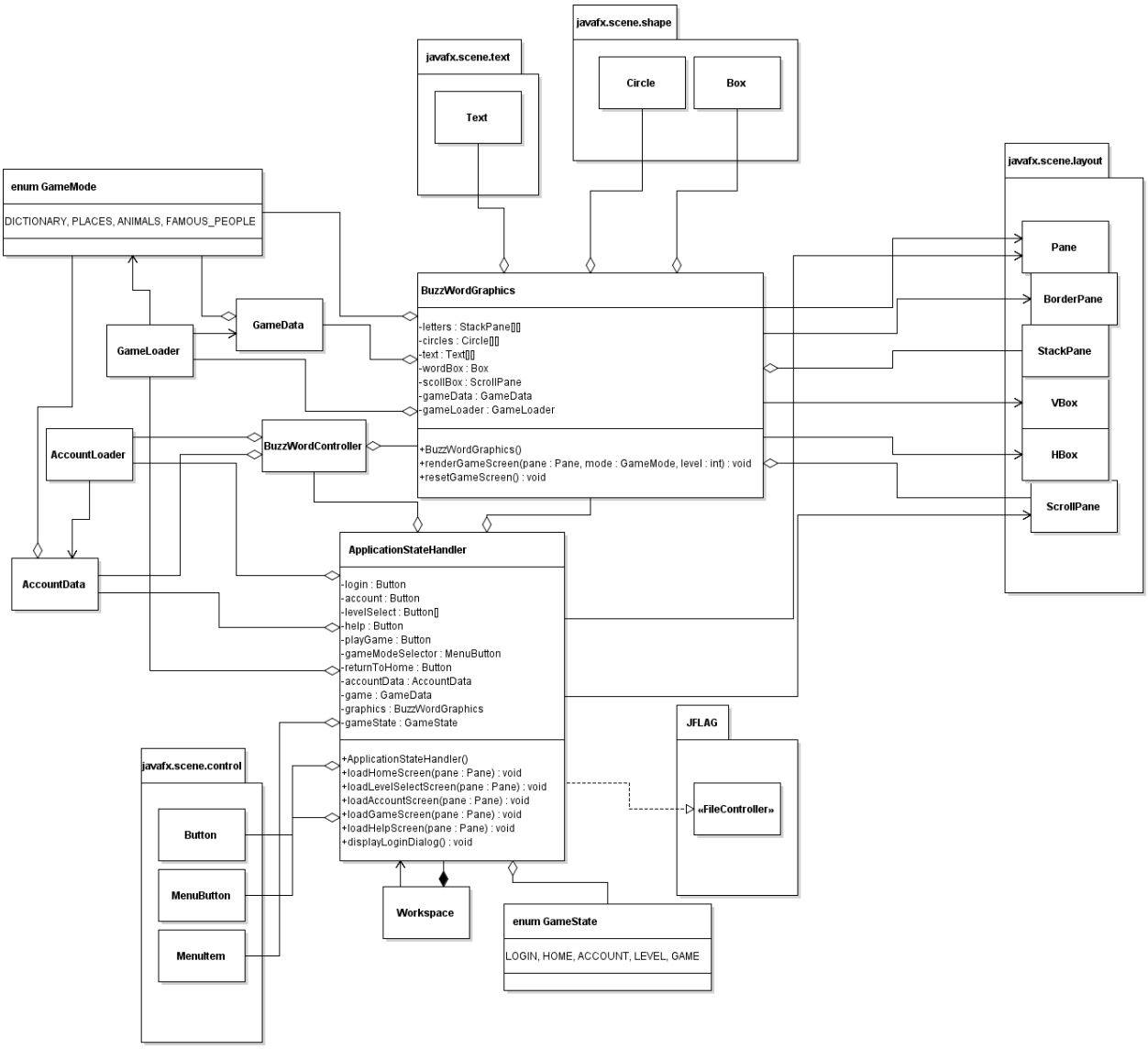

Figure 3.3: Detailed ApplicationStateHandler, BuzzWordGraphics, GameState, and GameMode UML Class Diagrams part of the GUI Package

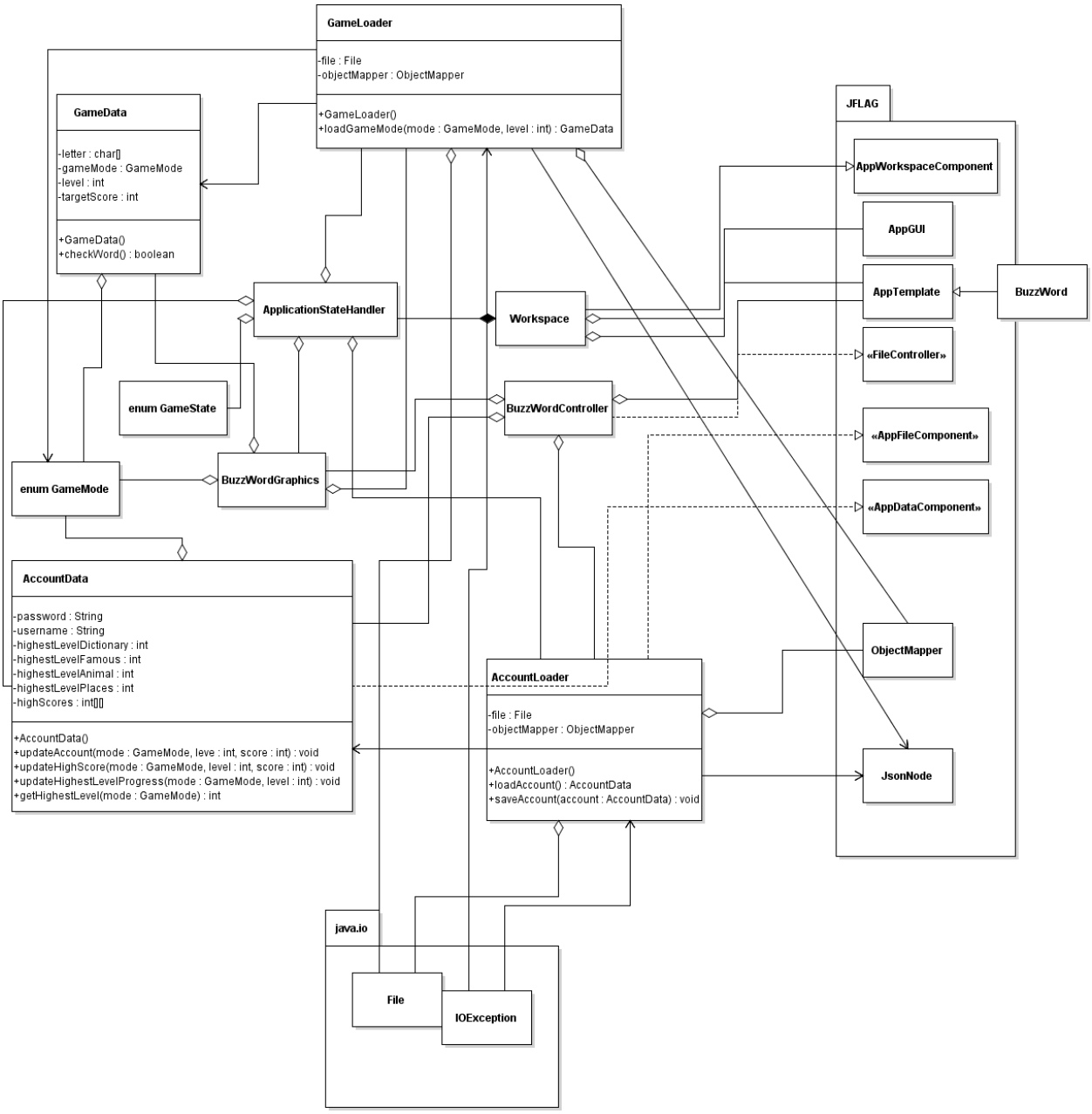

Figure 3.4: Detailed AccountData, AccountLoader, GameData, and GameLoader UML Class Diagrams part of the Data package

4 Method-Level Design Viewpoint

Now that the classes have been designed, how the data will flow and how the methods will be used need to be specified. The following UML Sequence Diagrams detail the methods to be developed which will be called for each event response.

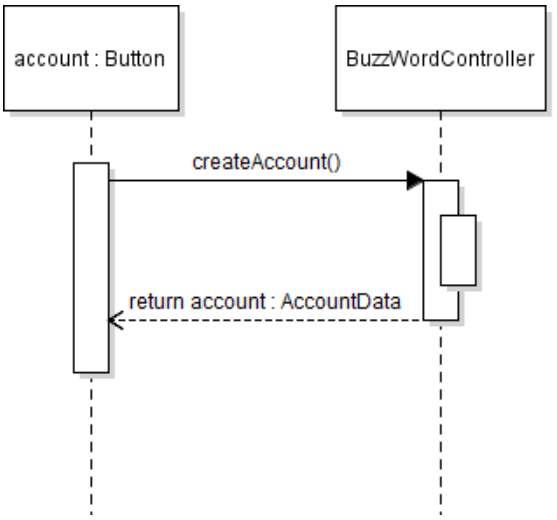

Figure 4.1: Create Profile UML Sequence Diagrams

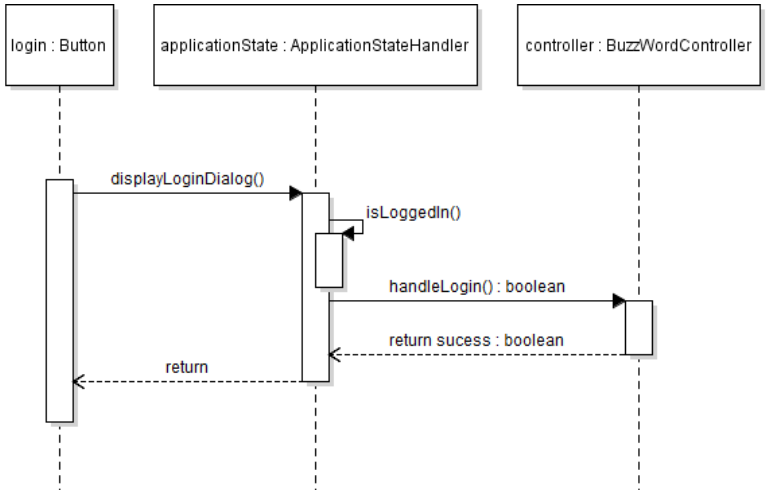

Figure 4.2: Login/Logout UML Sequence Diagrams

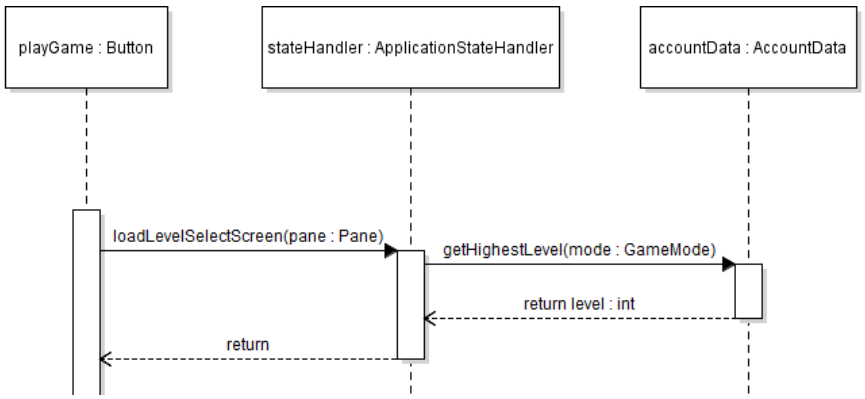

Figure 4.3: Start Playing UML Sequence Diagrams

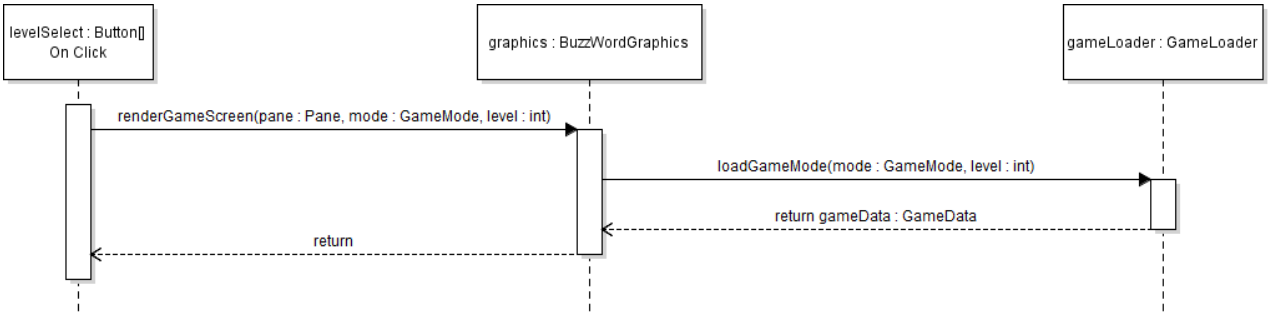

Figure 4.4: Select Level UML Sequence Diagrams

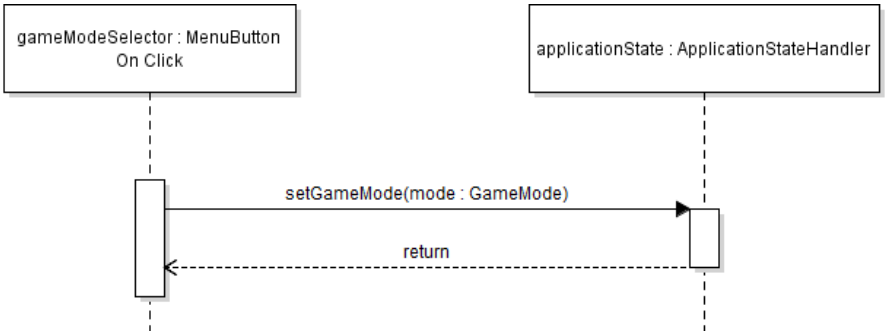

Figure 4.5: Select Game Mode UML Sequence Diagrams

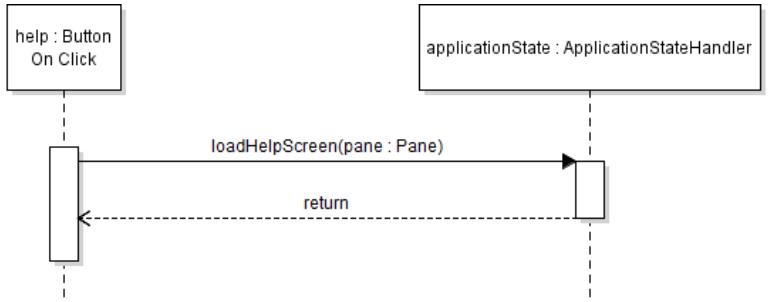

Figure 4.6: View Help UML Sequence Diagrams

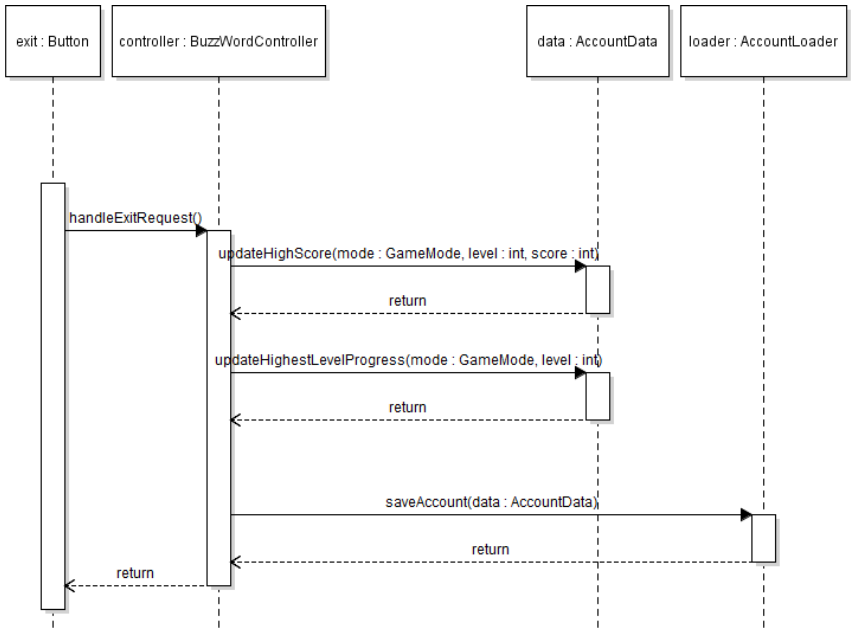

Figure 4.7: Quit Application UML Sequence Diagrams

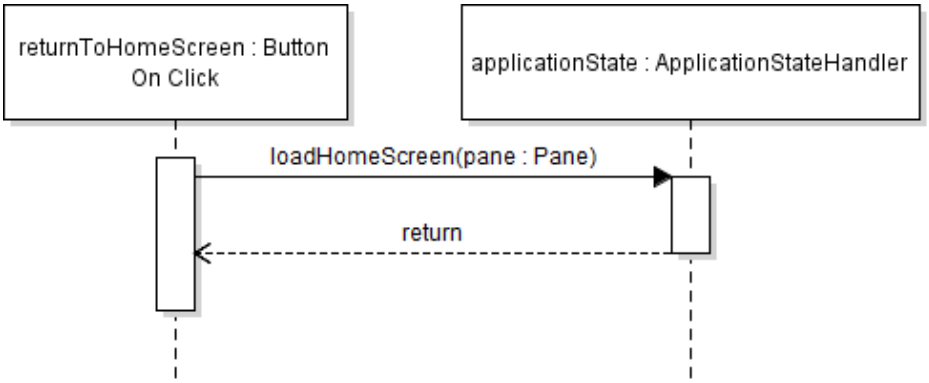

Figure 4.8: Return to Home Screen UML Sequence Diagrams

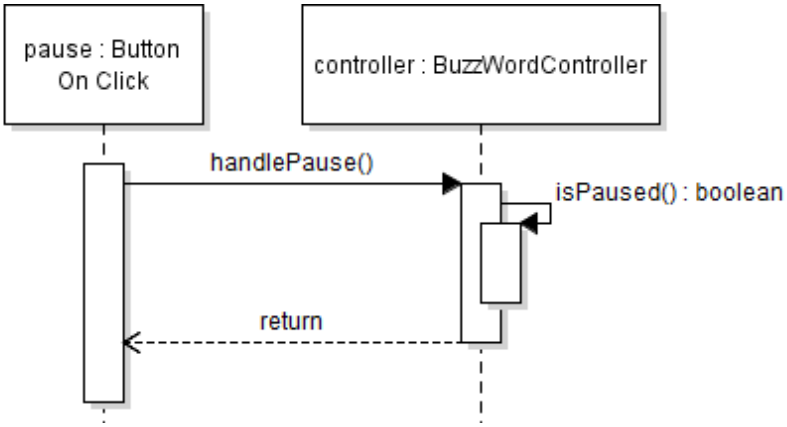

Figure 4.9: Pause/Resume Game UML Sequence Diagrams

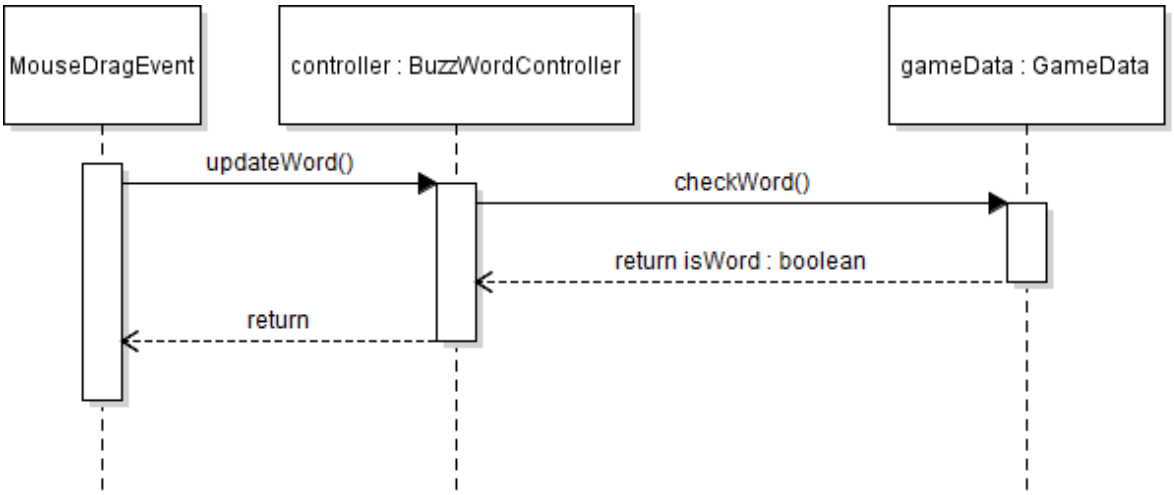

Figure 4.10: Select word by mouse drag UML Sequence Diagrams

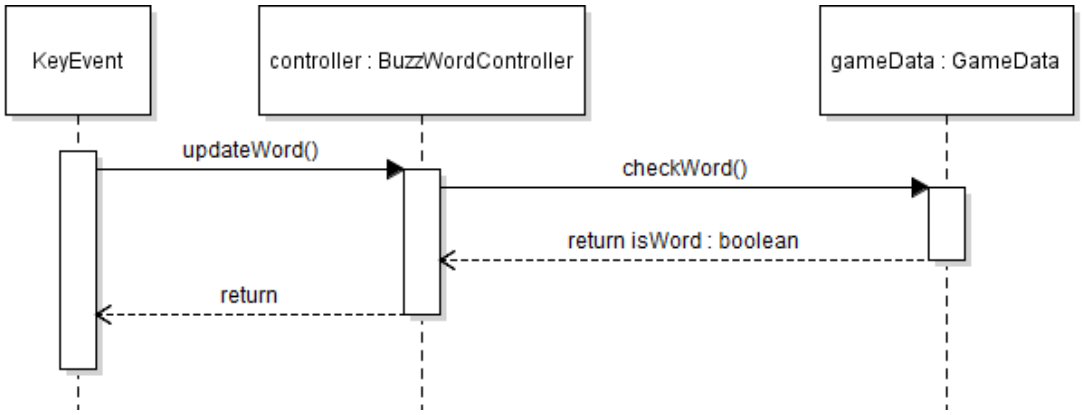

Figure 4.11: Select word by typing UML Sequence Diagrams

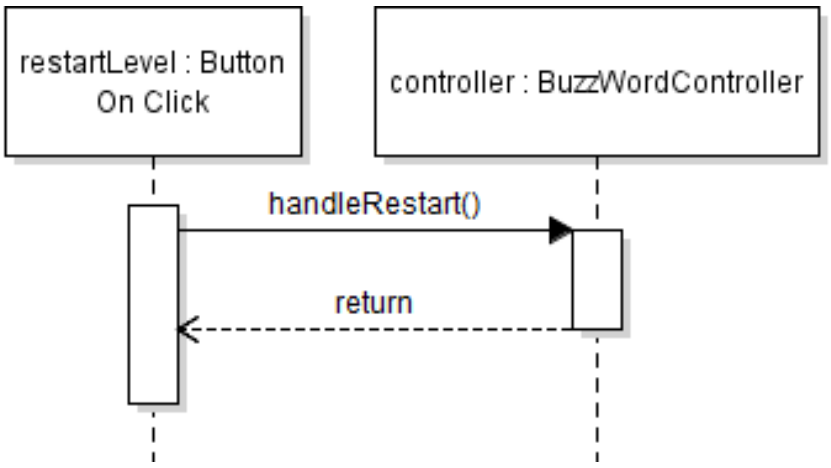

Figure 4.12: Replay Level UML Sequence Diagrams

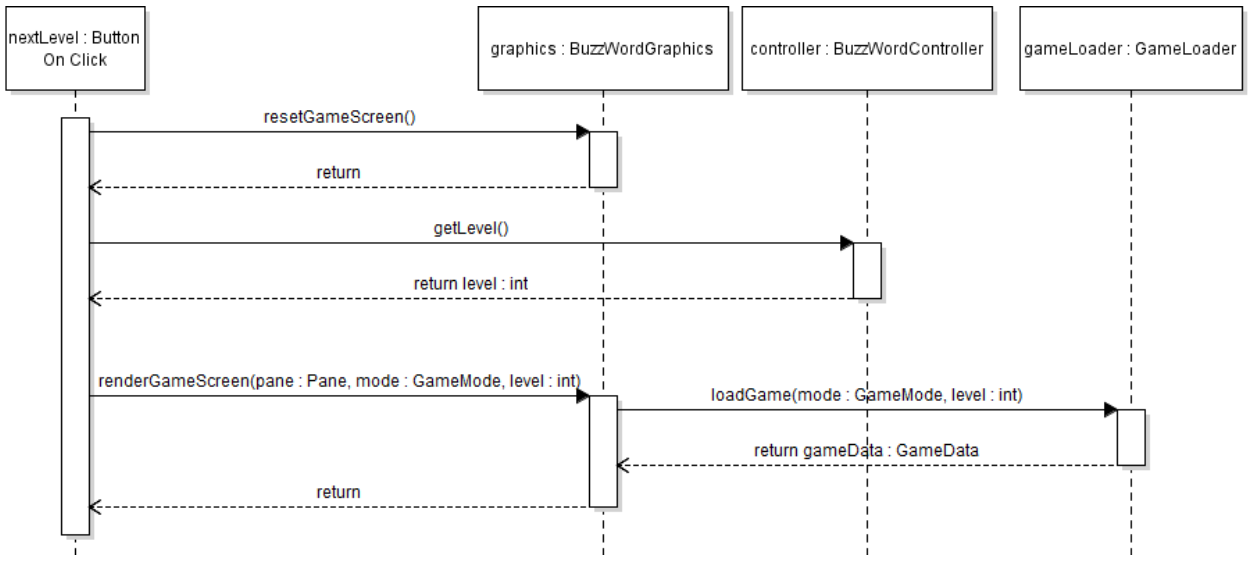

Figure 4.13: Start Next Level UML Sequence Diagrams

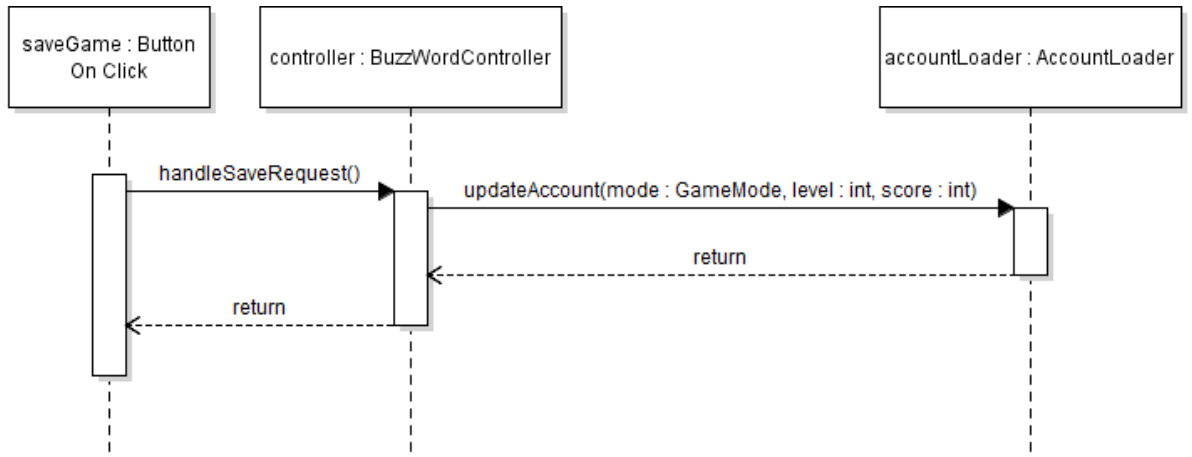

Figure 4.14: Save Progress UML Sequence Diagrams

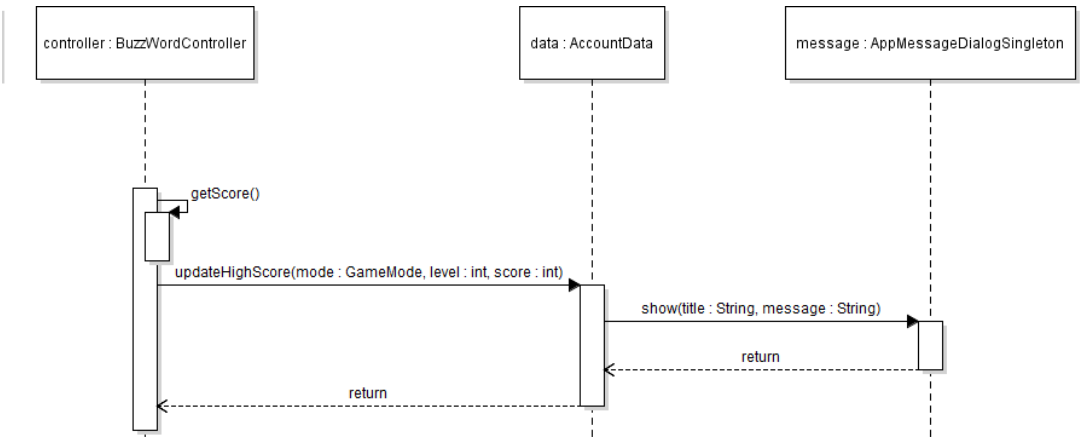

Figure 4.15: Close "Personal Best" Dialog UML Sequence Diagrams

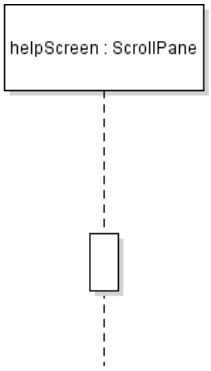

Figure 4.16: Scroll Through Help Sequence Diagrams

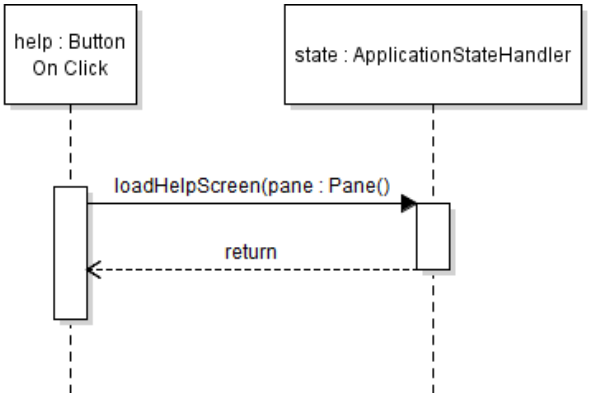

Figure 4.17: View/Edit Profile UML Sequence Diagrams

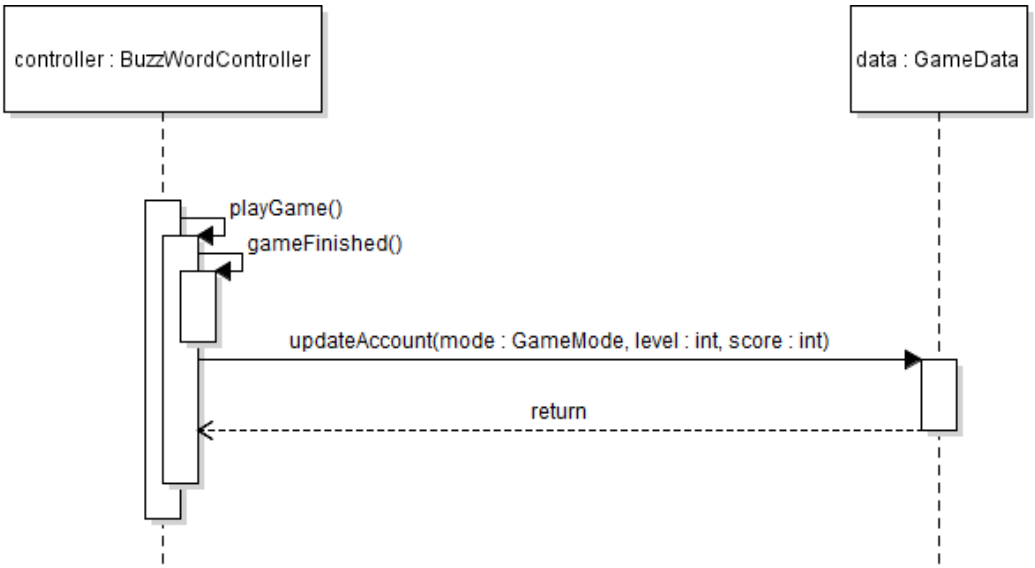

Figure 4.18: Level Gameplay Ends UML Sequence Diagrams

5 File Structure and Formats

The file structure of the project is shown in the figure below. The Jackson JSON API is saved as a JAR file and is placed separately from the JFLAG.

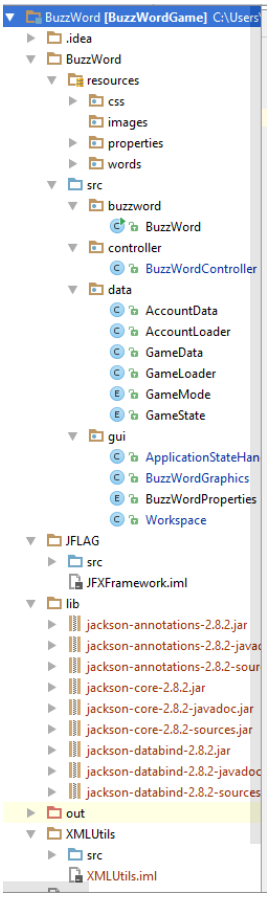

Figure 5.1: File Structure of the BuzzWord application in the IntelliJ IDEA IDE

5.1 BuzzWord File Structure

The game will utilize two JSON files. One for storing the game data about each game mode and each level. The other will be used for the account data such as username, password, and their high scores.

6 Supporting Information

This document will serve as a reference to help develop the code for BuzzWord. A table of contents is given to help find important sections.

6.1 Table of Contents

1 Introduction 2

1.1 Purpose 2

1.2 Scope 2

1.3 Definitions, Acronyms, and Abbreviations 2

1.4 References 4

1.5 Overview 4

2 Package-Level Design Viewpoint 4

2.1 Buzz Word Overview 5

2.2 Java API Usage 6

2.3 Java API Usage Descriptions 6

3 Class-Level Design Viewpoint 8

4 Method-Level Design Viewpoint 12

5 File Structure and Formats 18

5.1 BuzzWord File Structure 18

6 Supporting Information 19

6.1 Table of Contents 19

6.2 Appendixes 20

6.2 Appendixes

N/A