

Run Louie Run!

**By: Nabeel Vali, Andrew
Freiman, and Kehlsey Lewis**

**For: Dr. Jagadeesh Nandigam,
CIS-350-02**

Table of contents

Project Description	2
Release 2 & 1 features	2
Screenshots.....	3-6
Use Cases.....	7-15
UML Diagram.....	16-18
Checkstyle violations report.....	19-21
Find Bugs report.....	22-23
Git Log.....	24-25
Code Coverage Report.....	26-30
Roles, Responsibilities, Reflections.....	31-33

Project Description:

“Run Louie Run!” is an infinite runner game where the player controls Louie the Laker as he runs through Grand Valley’s campus all while collecting anchors and avoiding evil exams! The goal is to dodge all obstacles using the spacebar to jump, running as far as possible. If the player collides with an obstacle, the game is over and he or she must start again. Currently there is only one obstacle, an evil exam, but more content is planned for the second release.

Release 1 Features:

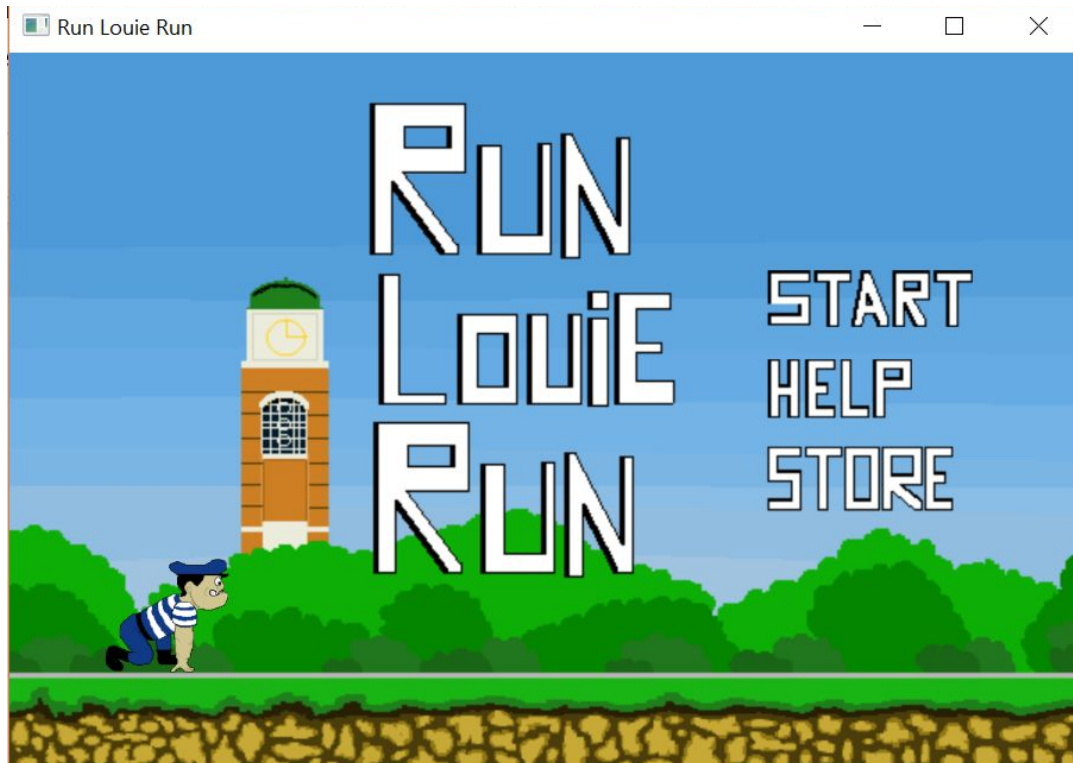
- Character movement:
 - Basic movements such as the running and jumping animations for the main character and enemies have been implemented.
- Sound effects:
 - The game theme music (main menu & running) and sound effects such as jumping, game over, and dying have been created and implemented.
- Landscape:
 - A static landscape portraying the GVSU campus is displayed in the background of the game.
- Obstacles:
 - Evil exams were designed and implemented to act as random obstacles for Louie to dodge
- User Interface:
 - A basic user interface that allows users to start the game, view the instructions, and return to the main menu or restart the game after the players death as been implemented.

Release 2 Features:

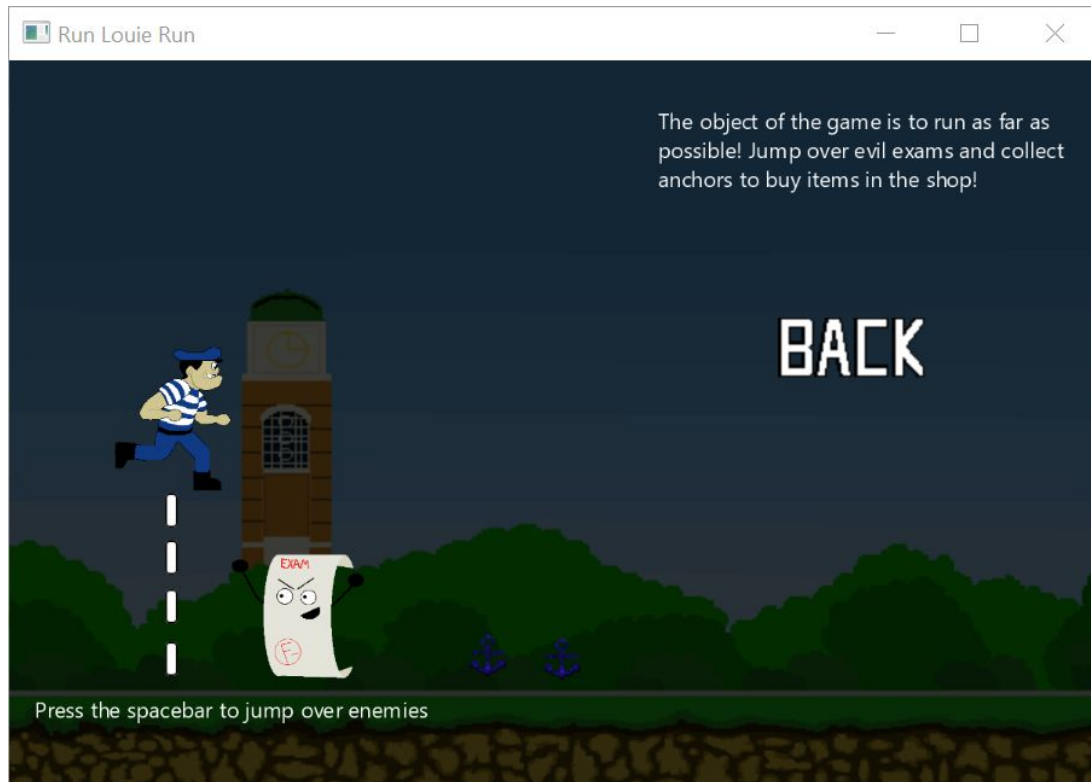
- Character customization:
 - Various outfits are available for Louie to wear.
- Score System:
 - The players score now displays during gameplay. Players can now collect “anchor” coins to purchase in the shop.
- In-Game Shop:
 - An in-game store was implemented to allow players to spend earned anchor coins to buy cosmetic modifications for Louie.
- Moving Landscape:
 - A moving landscape portraying the GVSU campus is displayed in the background of the game.
- Countdown:
 - A countdown clock and sound effects were added to the beginning of the game.

ScreenShots

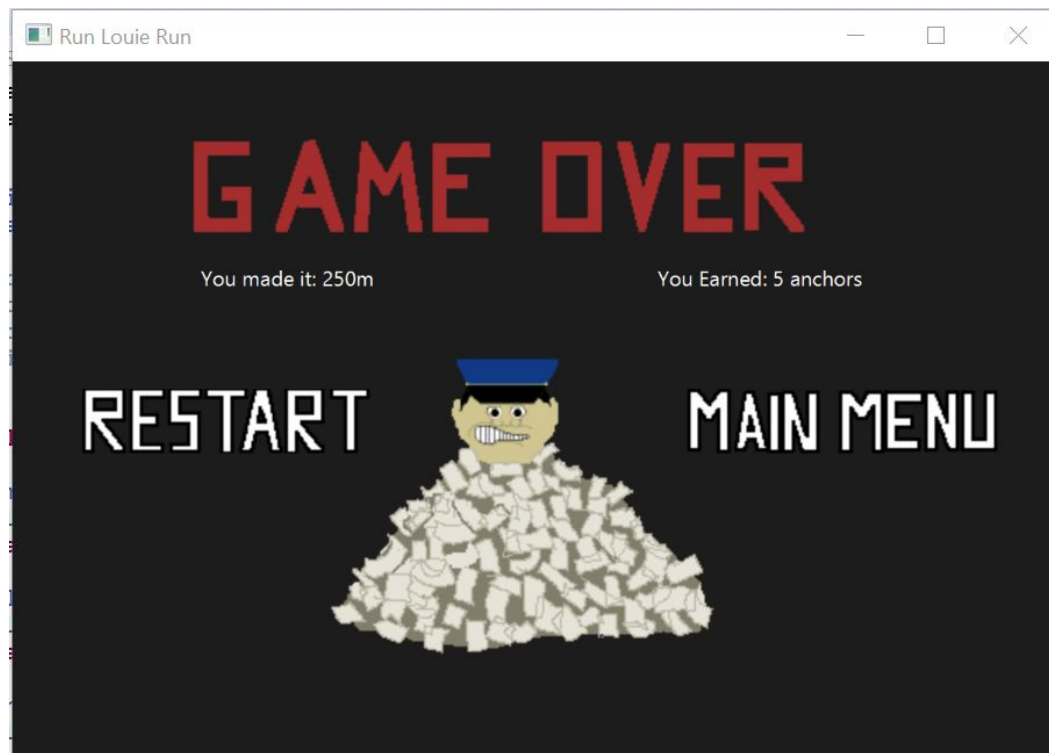
Game Menus



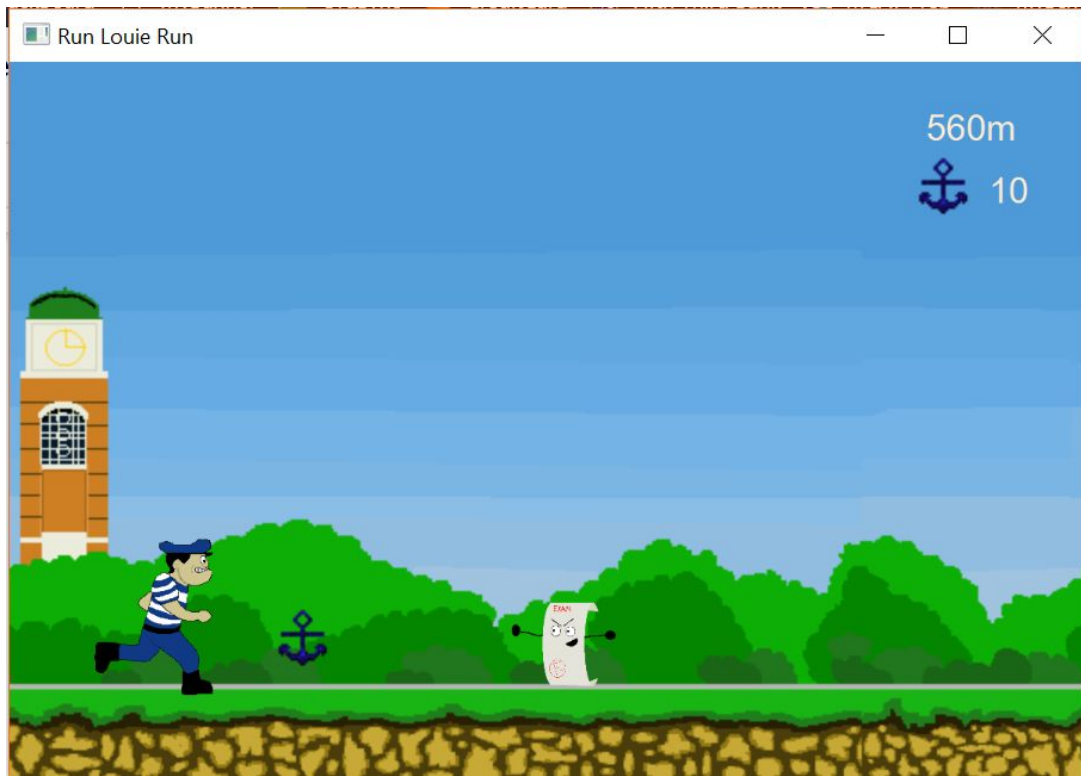
Main Menu

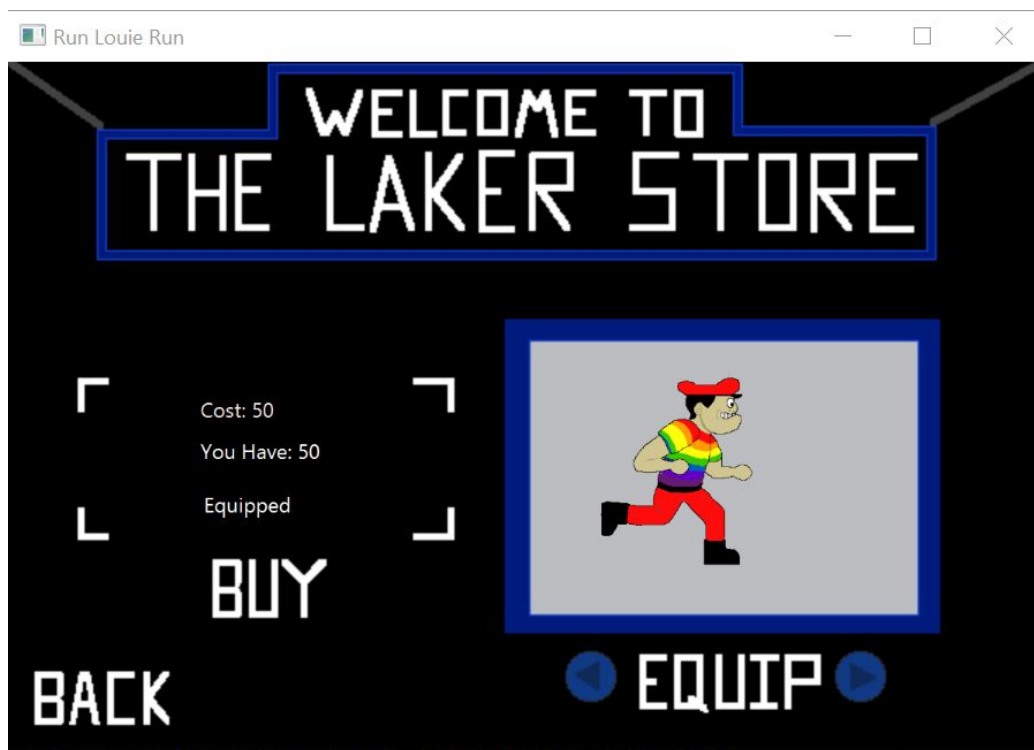
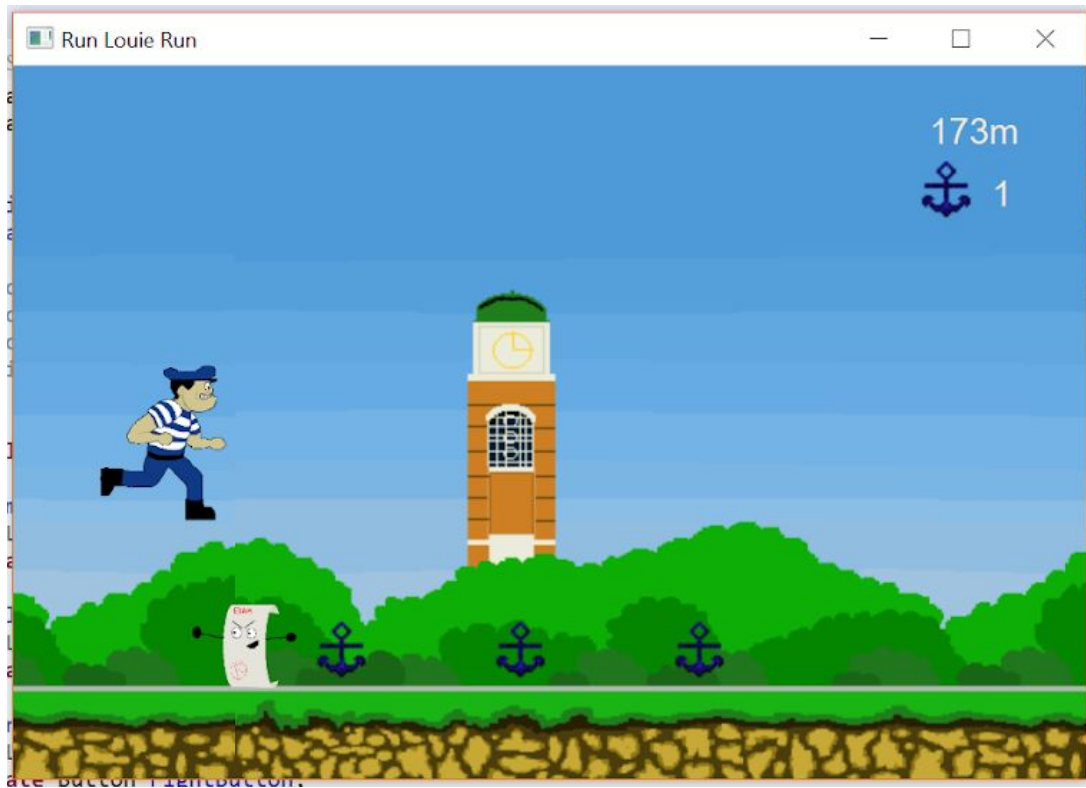


Help Menu

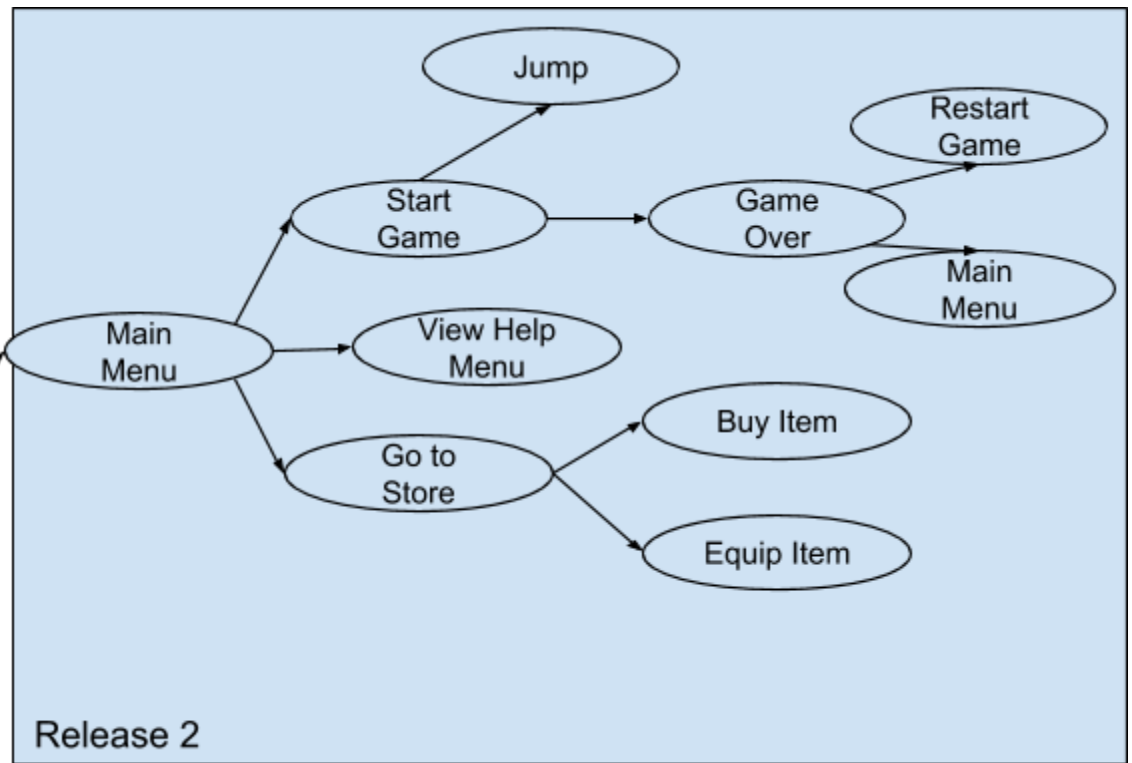


Game Over Menu





Use Cases



Use Case Descriptions:

Name	Start Game
ID	RLR-UC1
Brief Description	Describes behavior of start button
Actors	Primary - Player
Trigger(s)	Player selects the “Start” button
Preconditions	The game window is open and in the main menu
Primary Flow	<ol style="list-style-type: none">1. Player clicks start button2. The app opens the game window
Alternate Flows	<ul style="list-style-type: none">- Player does not press the start button- Player presses the wrong button and exits the game instead
Success Guarantees	The player plays the game
Minimal Guarantees	The player sees the game window

Name	Open Help Menu
ID	RLR-UC2
Brief Description	Describes how the user opens the help menu
Actors	Primary - Player
Trigger(s)	Player selects the “Help” button
Preconditions	The game window is open and in the main menu
Primary Flow	<ol style="list-style-type: none">1. Player selects help button2. The app opens the help window
Alternate Flows	<ul style="list-style-type: none">- Player does not press the help button- Player presses the wrong button and exits or starts the game instead
Success Guarantees	The player sees the game window

Minimal Guarantees	The player sees the help window
---------------------------	---------------------------------

Name	Restart Game
ID	RLR-UC3
Brief Description	Describes how the player can restart the game after losing
Actors	Primary - Player
Trigger(s)	Player selects the “Restart” button
Preconditions	The player has lost the game and the game over window is open
Primary Flow	<ol style="list-style-type: none"> 1. Player selects restart button 2. The app opens a new game window
Alternate Flows	<ul style="list-style-type: none"> - Player does not press the restart button - Player presses the wrong button and exits the game or returns to the main menu instead
Success Guarantees	The player restarts the game
Minimal Guarantees	The player sees the game over screen

Name	Go to Main Menu after losing
ID	RLR-UC4
Brief Description	Describes how the player can return to the main menu after losing
Actors	Primary - Player
Trigger(s)	Player selects the “Main Menu” button from the game over screen
Preconditions	The player has lost and the game over screen is open
Primary Flow	<ol style="list-style-type: none"> 1. Player clicks main menu button

	2. The app returns to the main menu window
Alternate Flows	<ul style="list-style-type: none"> - Player does not press the main menu button - Player presses the restart button and restarts the game instead - Player exits the game instead
Success Guarantees	The player returns to the main menu window
Minimal Guarantees	The player sees the game over screen

Name	Go to Main Menu from Store
ID	RLR-UC5
Brief Description	Describes how the player can return to the main menu from the store menu
Actors	Primary - Player
Trigger(s)	Player selects the “Back” button from the in-game store screen
Preconditions	The player is in the store.
Primary Flow	<ol style="list-style-type: none"> 1. Player clicks back button 2. The app returns to the main menu window
Alternate Flows	<ul style="list-style-type: none"> - Player does not press the back button - Player exits the game instead
Success Guarantees	The player returns to the main menu window
Minimal Guarantees	The player sees the in-game store window

Name	Jump
ID	RLR-UC6
Brief Description	Describes how the player can Jump To

	dodge an enemy
Actors	Primary - Player
Trigger(s)	Player has pressed the spacebar
Preconditions	The game has started and enemies are moving towards the player
Primary Flow	<ol style="list-style-type: none"> 1. Player sees enemies moving towards louie 2. Player presses space bar 3. Louie jumps into the air dodging the enemy 4. Louie lands back on the ground
Alternate Flows	<ul style="list-style-type: none"> - Player does not press the spacebar and dies - Player exits the game instead
Success Guarantees	The player jumps into the air dodging the enemy
Minimal Guarantees	The enemy collides with the player and causes a gameover scenario

Name	Game Over
ID	RLR-UC7
Brief Description	Describes how the player Causes a gameover state
Actors	Primary - Player
Trigger(s)	Player has collided with an enemy
Preconditions	Game has started and enemies are moving towards the player
Primary Flow	<ol style="list-style-type: none"> 1. Game has started 2. Player see's enemy moving towards player 3. Player collides with enemy 4. Game over screen shows
Alternate Flows	<ul style="list-style-type: none"> - Player exits the game

Success Guarantees	Game over screen appears prompting user to restart or visit the main menu
Minimal Guarantees	The enemy collides with the player and game freezes

Name	Open The Laker Store
ID	RLR-UC8
Brief Description	Describes how the player interacts with the game to open the Laker Store
Actors	Primary - Player
Trigger(s)	Player clicks “Store” button
Preconditions	The game window is open and in the main menu
Primary Flow	<ol style="list-style-type: none"> 1. Player clicks the “Store” button 2. Player sees the store window 3. Player maneuvers through store to see item choices using the arrow buttons
Alternate Flows	<ul style="list-style-type: none"> - Player clicks wrong button and exits the game, starts the game, or sees help menu
Success Guarantees	The player maneuvers through store window
Minimal Guarantees	The player views the store window

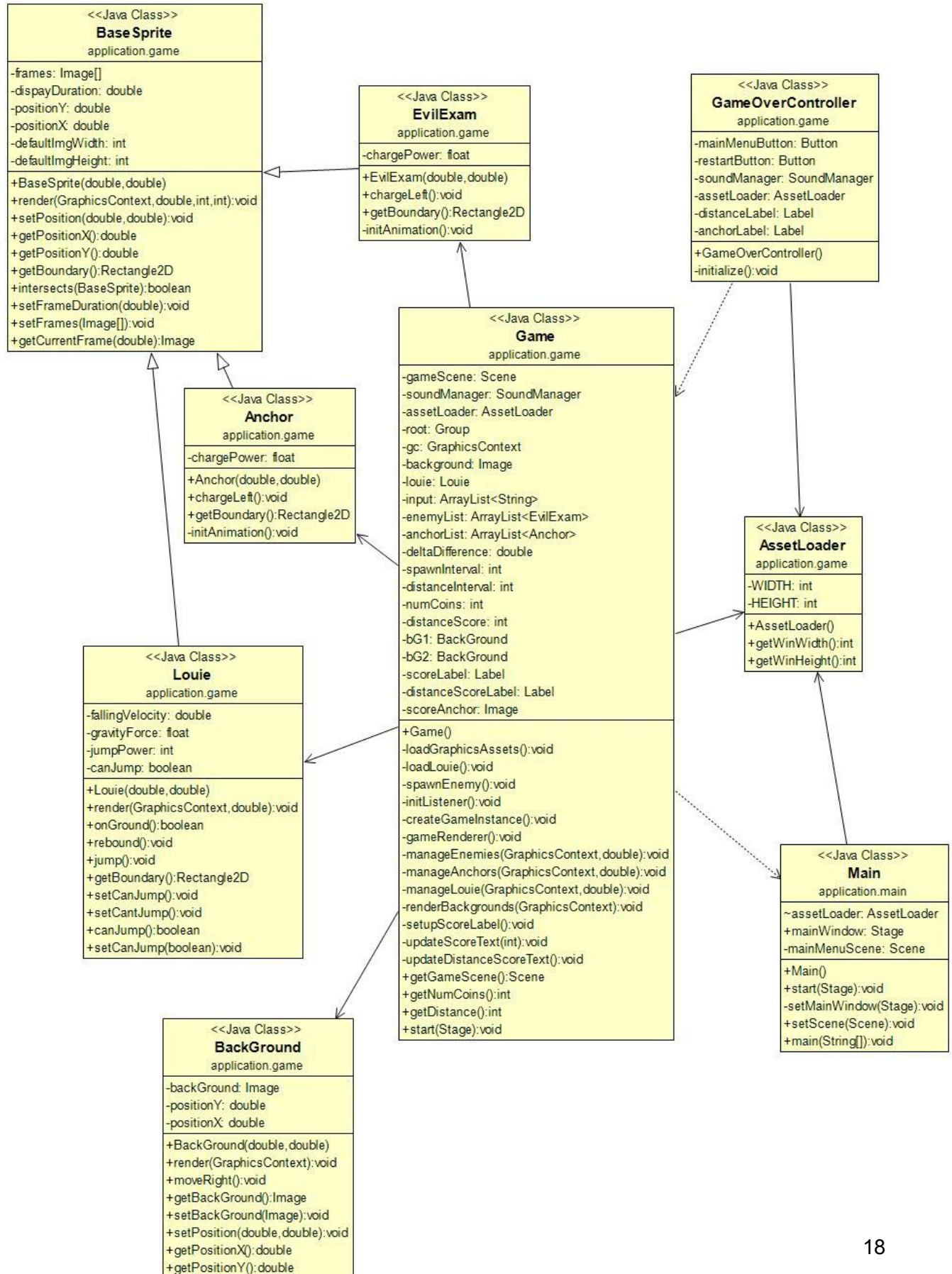
Name	Buy an Item
ID	RLR-UC9
Brief Description	Describes how the player interacts with the Laker Store to purchase an item

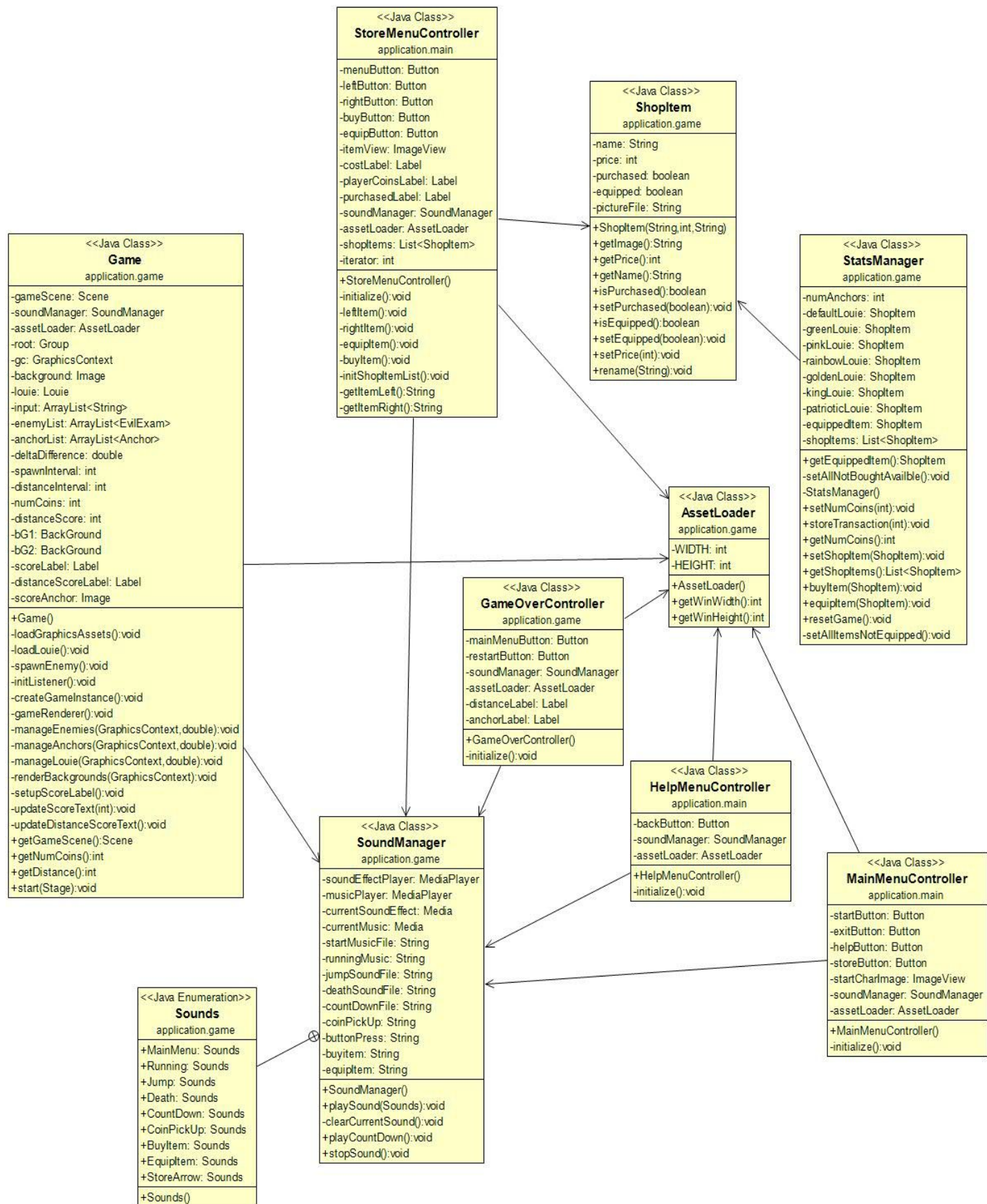
Actors	Primary - Player
Trigger(s)	Player clicks the “buy” button on selected item
Preconditions	The store window is open
Primary Flow	<ol style="list-style-type: none"> 1. Player clicks the “buy” button for the desired item 2. Bought item is equipped 3. Number of coins decreases by item price
Alternate Flows	<ul style="list-style-type: none"> - Player does not have enough coins to purchase the item. “Insufficient funds” displays on window. - Player already owns the item. “Equip Available” is displayed
Success Guarantees	The player buys the item
Minimal Guarantees	The player learns if they have enough coins to buy the item

Name	Equip an Item
ID	RLR-UC10
Brief Description	Describes how the player interacts with the Laker Store to equip an already purchased item
Actors	Primary - Player
Trigger(s)	Player has clicks the “Equip” button under the desired item
Preconditions	The store window is open
Primary Flow	<ol style="list-style-type: none"> 1. Player clicks the “Equip” button 2. The item equips and “Equipped” text displays in window
Alternate Flows	<ul style="list-style-type: none"> - Player has not purchased the item yet and “not purchased” text is displayed

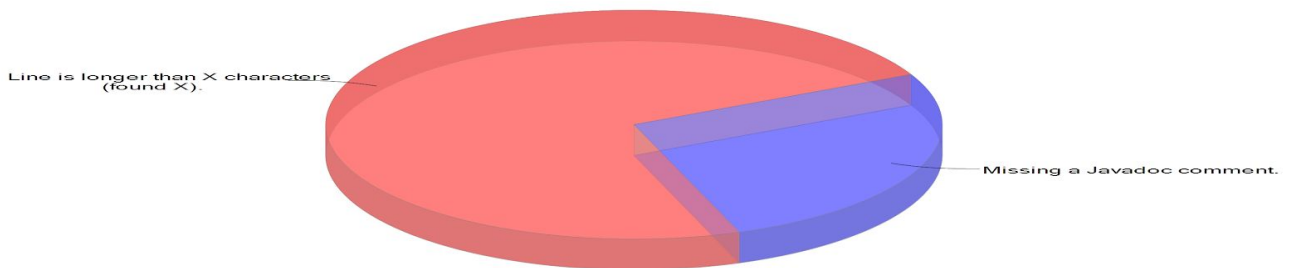
	<ul style="list-style-type: none"> - Item is already equipped and nothing changes.
Success Guarantees	The player equips the desired purchased item
Minimal Guarantees	The player learns if they are able to equip the desired item

UML Diagram





Checkstyle Violations Report and Chart



The above captions pointing to the graph have the same violations as the ones in the chart below.

Problems

Bug Explorer

Console

Checkstyle violations

Coverage

Call Hierarchy

Checkstyle violations chart

Overview of Checkstyle violations - 15 markers in 2 categories (Filter matched 15 of 15 items)

Checkstyle violation type	Marker count
Line is longer than X characters (found X).	11
Missing a Javadoc comment.	4

Problems	Bug Explorer	Console	Checkstyle violations	Coverage	Call Hierarchy	Checkstyle violations chart
Details of Checkstyle violation "Line is longer than X characters (found X)." - 11 occurrences						
Resource	In Folder	Line	Message			
Game.java	/RunLouieRun/src/application/Game	199	Line is longer than 80 characters (found 82).			
SoundManager.j...	/RunLouieRun/src/application/Game	65	Line is longer than 80 characters (found 82).			
SoundManager.j...	/RunLouieRun/src/application/Game	77	Line is longer than 80 characters (found 82).			
SoundManager.j...	/RunLouieRun/src/application/Game	79	Line is longer than 80 characters (found 82).			
EvilExam.java	/RunLouieRun/src/application/Game	52	Line is longer than 80 characters (found 84).			
SoundManager.j...	/RunLouieRun/src/application/Game	85	Line is longer than 80 characters (found 84).			
Game.java	/RunLouieRun/src/application/Game	114	Line is longer than 80 characters (found 85).			
Game.java	/RunLouieRun/src/application/Game	210	Line is longer than 80 characters (found 87).			
Game.java	/RunLouieRun/src/application/Game	211	Line is longer than 80 characters (found 88).			
GameOverContr...	/RunLouieRun/src/application/Game	54	Line is longer than 80 characters (found 88).			
Game.java	/RunLouieRun/src/application/Game	212	Line is longer than 80 characters (found 96).			

Problems

Bug Explorer

Console

Checkstyle violations

Coverage

Call Hierarchy

Checkstyle violations chart

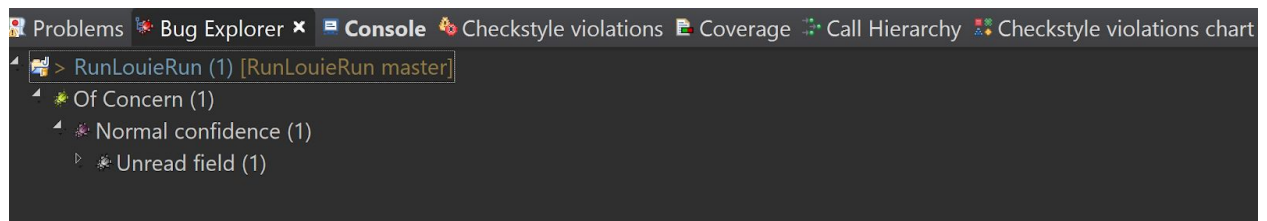
Details of Checkstyle violation "Missing a Javadoc comment." - 4 occurrences

Resource	In Folder	Line	Message
SoundManager.j...	/RunLouieRun/src/application/Game	48	Missing a Javadoc comment.
SoundManager.j...	/RunLouieRun/src/application/Game	49	Missing a Javadoc comment.
SoundManager.j...	/RunLouieRun/src/application/Game	50	Missing a Javadoc comment.
SoundManager.j...	/RunLouieRun/src/application/Game	51	Missing a Javadoc comment.

We began with over 200 checkstyle exceptions, narrowing them down to 15. The current exceptions include lines being longer than 80 characters, we made an effort to fix most, leaving only the ones that exceeded 80 by a trivial amount. Additionally, checkstyle claims that one of

our enumerated types does not have a javadoc annotation, yet one is presently commented right above the enumerated type.

Find Bugs Bug Report



We initially started with 10 bugs and narrowed it down to one, currently we have an boolean variable that the program claims is not being manipulated, however, throughout the program there are calls to change the boolean from true to false, so we decided to ignore this notification.

Git Log

URL to Git Repository:







<https://github.com/Nabz786/RunLouieRun>

Javadoc API:






<https://nabz786.github.io/RunLouieRun/>

Code Coverage Report






Code Coverage for all classes in Game Package.

Coverage					
Session: Main (Feb 27, 2018 7:09:01 PM)					
Counter	Coverage	Covered	Missed	Total	
Instructions	 99.0 %	394	4	398	
Branches	 93.8 %	15	1	16	
Lines	 97.8 %	90	2	92	
Methods	 92.9 %	13	1	14	
Types	 100.0 %	4	0	4	
Complexity	 90.9 %	20	2	22	







Game

Coverage					
Session: Main (Feb 27, 2018 7:09:01 PM)					
Counter	Coverage	Covered	Missed	Total	
Instructions	 94.7 %	54	3	57	
Lines	 94.1 %	16	1	17	
Methods	 100.0 %	4	0	4	
Types	 100.0 %	1	0	1	
Complexity	 100.0 %	4	0	4	







Game Over

Coverage					
Session: Main (Feb 27, 2018 7:16:09 PM)					
Counter	Coverage	Covered	Missed	Total	
Instructions	 92.9 %	39	3	42	
Lines	 93.3 %	14	1	15	
Methods	 100.0 %	3	0	3	
Types	 100.0 %	1	0	1	
Complexity	 100.0 %	3	0	3	






Help Menu

Coverage				
Session: Main (Feb 27, 2018 7:09:01 PM)				
Counter	Coverage	Covered	Missed	Total
Instructions	 100.0 %	63	0	63
Branches	 100.0 %	2	0	2
Lines	 100.0 %	15	0	15
Methods	 100.0 %	4	0	4
Types	 100.0 %	1	0	1
Complexity	 100.0 %	5	0	5







Louie Sprite

Coverage				
Session: Main (Feb 27, 2018 7:09:01 PM)				
Counter	Coverage	Covered	Missed	Total
Instructions	 97.6 %	205	5	210
Branches	 80.0 %	4	1	5
Lines	 100.0 %	45	0	45
Methods	 88.9 %	8	1	9
Types	 100.0 %	2	0	2
Complexity	 84.6 %	11	2	13

Sound Manager






Coverage				
Session: Main (Feb 27, 2018 7:09:01 PM)				
Counter	Coverage	Covered	Missed	Total
Instructions	 84.3 %	75	14	89
Lines	 92.3 %	24	2	26
Methods	 90.0 %	9	1	10
Types	 100.0 %	1	0	1
Complexity	 90.0 %	9	1	10

Base Sprite






Counter	Coverage	Covered	Missed	Total
Instructions	 100.0 %	63	0	63
Branches	 100.0 %	2	0	2
Lines	 100.0 %	15	0	15
Methods	 100.0 %	4	0	4
Types	 100.0 %	1	0	1
Complexity	 100.0 %	5	0	5

Evil Exam



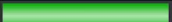
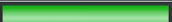

Code Coverage for all classes in Main Package

Coverage				
Session: Main (Feb 27, 2018 7:18:32 PM)				
Counter	Coverage	Covered	Missed	Total
Instructions	 100.0 %	44	0	44
Lines	 100.0 %	16	0	16
Methods	 100.0 %	5	0	5
Types	 100.0 %	1	0	1
Complexity	 100.0 %	5	0	5

Main Class













Coverage				
Session: Main (Feb 27, 2018 7:18:32 PM)				
Counter	Coverage	Covered	Missed	Total
Instructions	 94.8 %	55	3	58
Lines	 95.2 %	20	1	21
Methods	 100.0 %	4	0	4
Types	 100.0 %	1	0	1
Complexity	 100.0 %	4	0	4

Main Menu Controller

Coverage					
Session: Main (Feb 27, 2018 7:16:09 PM)					
Counter	Coverage	Covered	Missed	Total	
Instructions	 92.9 %	39	3	42	
Lines	 93.3 %	14	1	15	
Methods	 100.0 %	3	0	3	
Types	 100.0 %	1	0	1	
Complexity	 100.0 %	3	0	3	

Help Menu

Overview of Coverage

Main (Feb 27, 2018 7:18:32 PM)				
Element	Coverage	Covered Instructions	Missed Instructions	Total Instructions
src	 96.9 %	1,006	32	1,038
application.Main	 95.8 %	138	6	144
HelpMenuController.java	 92.9 %	39	3	42
Main.java	 100.0 %	44	0	44
MainMenuController.java	 94.8 %	55	3	58
application.Game	 97.1 %	868	26	894
GameOverController.java	 94.7 %	54	3	57
EvilExam.java	 100.0 %	63	0	63
BaseSprite.java	 84.3 %	75	14	89
Louie.java	 100.0 %	77	0	77
SoundManager.java	 97.6 %	205	5	210
Game.java	 99.0 %	394	4	398

For the entire program we attained a coverage percentage of 96.9%.

Roles, Responsibilities, Reflections

- **Nabeel Vali:** Wrote majority of backend code, created basic front end, incorporated game assets created by team members into game, formatted with checkstyle, utilized spot bugs.
- **Kehlsey Lewis:** Created game artwork, coded main menu, help menu, game over menu, game countdown, created different Louie customizations, created UML diagrams, and Javadoc.
- **Andrew Freiman:** Created all game music (main menu theme, running theme), sound effects (jump, death, store sounds). Wrote the use cases, helped with sound integration into game. Worked on the in-game store code. Made store JUnit. Generated Javadoc for release 2.

Self Reflection by Each Team Member:

- **Nabeel Vali:** Overall our team had a great dynamic, with each member bring a diverse set of skills to the project. Each member of the group was proactive, staying in constant communication and making sure to contribute in a meaningful way. Throughout the development period of release one, everyone used their strengths to create unique content, never shying away from getting their hands dirty. From the code to documentation, each team member was fully adaptable taking on tasks and handling issues as required. It is important to note that our team met at least once a week and that all team members were willing to help each other with problems that we encountered. We all have many creative ideas and are looking forward to getting started on release 2.
- **Kehlsey Lewis:** I believe we worked well as a team together. We started working on release 1 as soon as we formed our group. We met at least once a week to discuss our ideas and fix any issues with our game. I feel everybody contributed equally and had an important part in the making the game. We each took advantage of our individual skills and applied them to our contribution in the project.
- **Andrew Freiman:** This was one of the best group project experiences I have had in my undergraduate education. We worked very well together. We met at least once weekly and stayed in constant communication through a group message. We really maximized on each of our strengths to complete the project and helped each other out when it came to our weaknesses. After we demoed release 1, we immediately got together to talk about our plans for release 2. We

divided up the tasks appropriately and we each got our part done in a timely fashion, helping each other out as needed.