Phase 4 Report

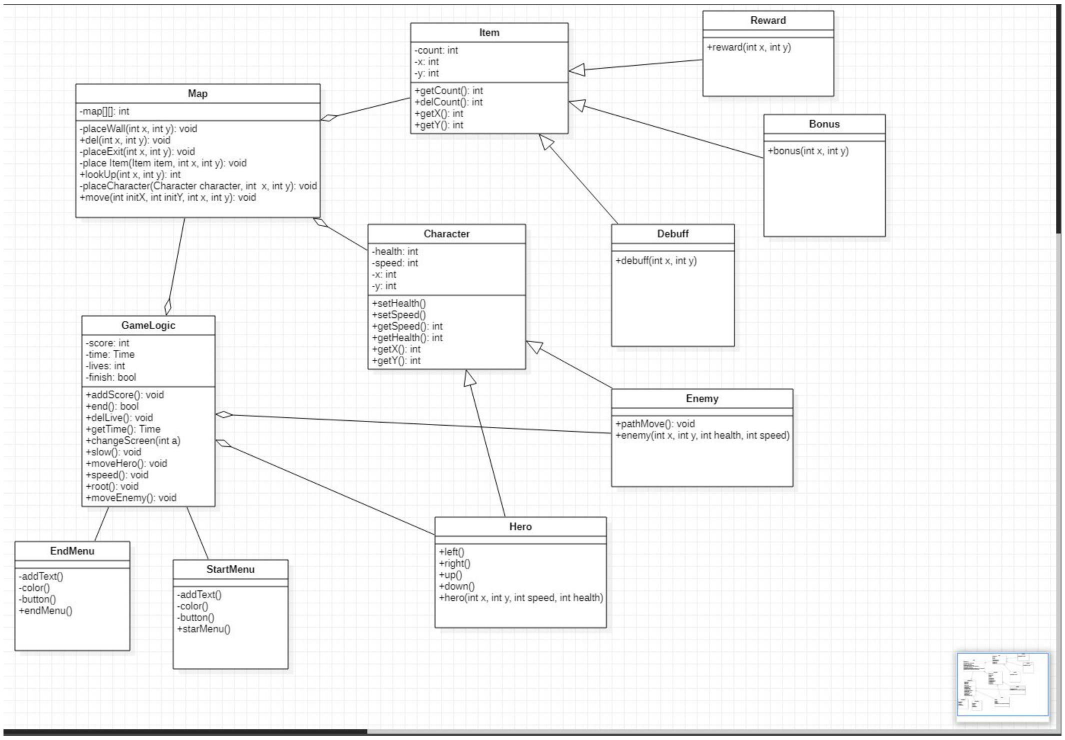
Jovan Bapla, Dennis Zhang
Josh Amato, Kris Zhang

Game Description

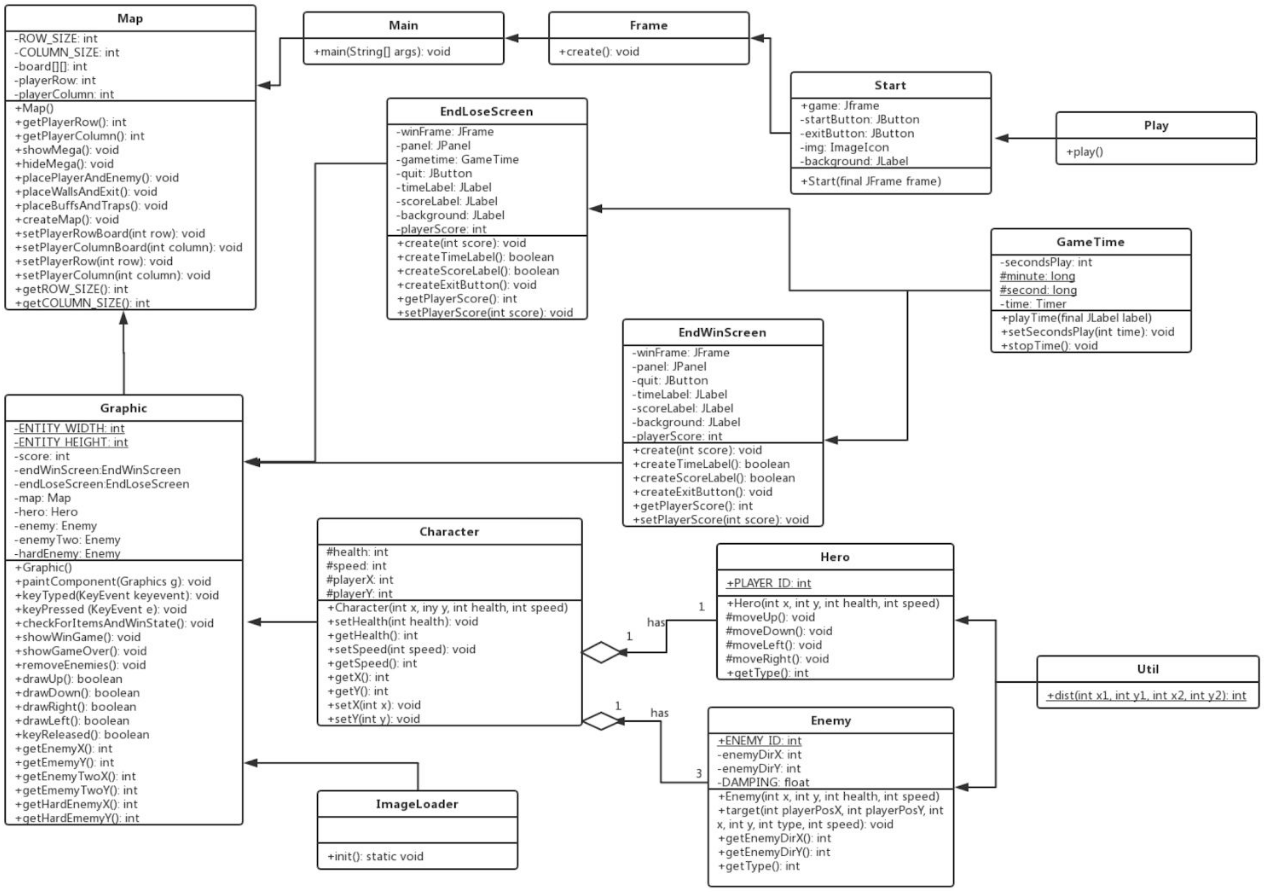
Cheezy Nibbles is a very simple arcade game. The form of the game is similar to the popular game pacman. The goal is for you, the player, to reach the portal on the other side of the open field without getting caught. Remember, you are being timed, so be as quick as you can. In order to open the portal, you must collect all the cheese slices scattered across the map. While collecting the cheese, beware of the bats, the owl and the mouse traps on the board. If you are within distance with the bats or the owl, you must escape and avoid collision or else the game is over. However, if you run over a trap, then only your score is affected but if your score reaches zero then the game will end. Other than the cheese slices, there is a bonus collectible on the board which appears randomly and boosts your score tremendously, but is not required for opening the portal!

Changes and Similarities

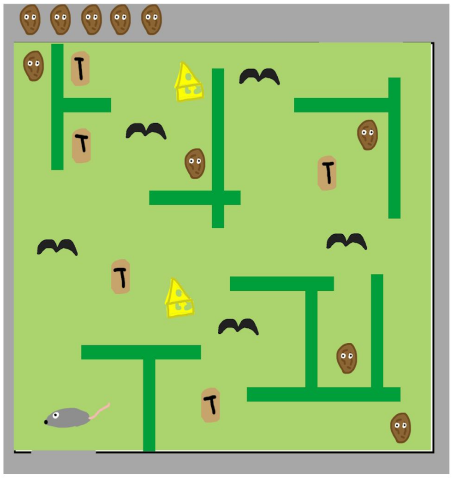
Original UML



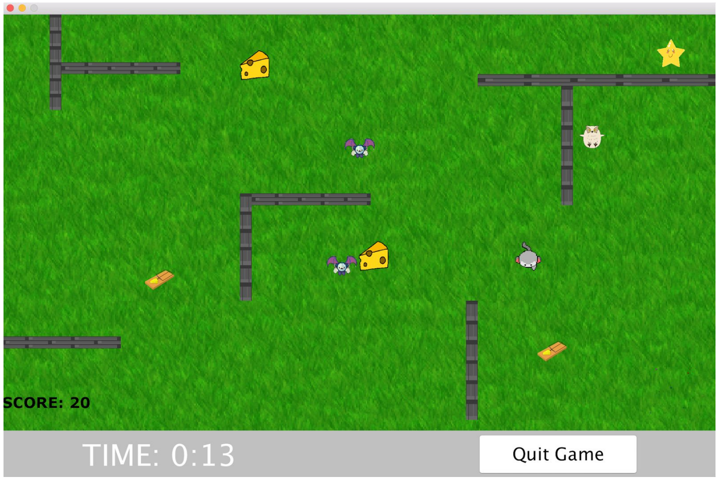
Final UML



Initial Design



Final Design



Code changes

Comparing the initial UML diagram and the final UML diagram, you can see the expansion of the final UML design as it incorporates many new additional classes and extended methods from the original UML design. A lot of the new inclusions came from composing the user interface component of the game. When constructing the actual classes during the production phase, there were noticeably new design issues, so in order to fix them, we needed to introduce new methods and practice separation of concerns. This was done by assigning original methods from the initial UML design and creating classes for them instead. There were definitely a lot more changes in the code for considerably larger classes, but was fairly faithful towards the smaller classes. This is because the smaller classes made more sense and had unquestionably no dependencies among other classes. With respect to the Item class and bonus class, rather than having them in their own classes, it was settled to have them implemented into which the team has established the map class as a core for handling all associated players and entity placements.

Aside from the StartMenu and EndMenu, the other components were not accounted for initially. The ImageLoader class was created in order to import all the images that are used in the game. Instead of having one EndMenu class, this was split into two distinct classes. One class was used for when the player wins the game (EndWinScreen) and the other is used for when the player loses the game (EndLoseScreen). The biggest addition that is not related to the user interface would be the GameTime class. This was created to have an internal timer in the game and would be a perfect extension for the end screens. The graphics class quickly became the replacement of the logic class because it would be simpler to include logic into where the game graphics move and have it be a hub class where everything comes together to create the game. There is also a loop inside that refreshes the frame so there already had to be logic statements inside the loop for the graphics to move.

Further enhancements included additional getters and setters methods and enclosing several code into appropriate methods. This augments the ability to efficiently test more of the production code during the testing phase and provide more ease when debugging. Having the ability to write testable code is an asset as it benefits for last minute trivial changes and essentially cost-free.

Overall the final UML diagram largely follows the general flow of the initial UML diagram. Some of the classes in the first diagram were expanded on, which then created new classes. The addition of the new classes was necessary as it helped encapsulate the code and allowed for better control of the code.

Concept changes

The team also talked about having multiple levels with a random map generator and buffs that offered the player different bonuses. However, it was decided that we should focus on just one level and make that as best as we could and if time permitted, create more levels in addition to that. For the buffs, we opted out of creating multiple bonuses that would affect the hero's stats, mainly for the same reasons as the map. We didn't want to add more features and potentially have them complicate things in the future. This is why we decided to have the cheese and the bonus reward both increase the player's score.

In terms of the actual look and feel of the game, the initial UI mockup and the actual game are almost identical. The game board itself is almost the same with the exception of a few icons missing and the addition of a timer, score, and a quit button in the final version.

Key Learning Points

The first learning experience was how to work with a team. This meant engaging in group meetings and having well organized plans that would work well with all members of the team. One of the biggest learning curves that benefited everyone in the group is being able to blend contemporary strengths and build trust among each other. The other one is encouraging group individuals to teach themselves how to improve their conflict resolution skills.

Having come to the end of the project, it is easy to say now that the first phase was just as important as the second and third phases. A lot of the roadblocks we encountered in the implementation phase and the testing phase was due to design issues from the first phase. Making sure you have great design from the beginning, accounting for even the smallest details, helps a lot in the long run.

On top of having great design initially, making sure you have great coding practices is just as important. Having clean and understandable code not only helps other team members look at your code but it helps with testing your code as well. It helps with diagnosing bugs since you'll have a good idea at where to look if a bug does appear. Part of having understandable code is including comments where it is necessary. Whether it is regular or JavaDocs, it is important to do them along the way and not leave them at the end where they might be forgotten.

Everything builds on itself. So you really have to make sure that at every step of the way, all the work that is being done is quality or it might have repercussions down the road. These repercussions can also keep building on and on to a point where it becomes too costly to change because there is heavy dependency on earlier fixes to something that should have been correctly implemented from the beginning.

Artifacts

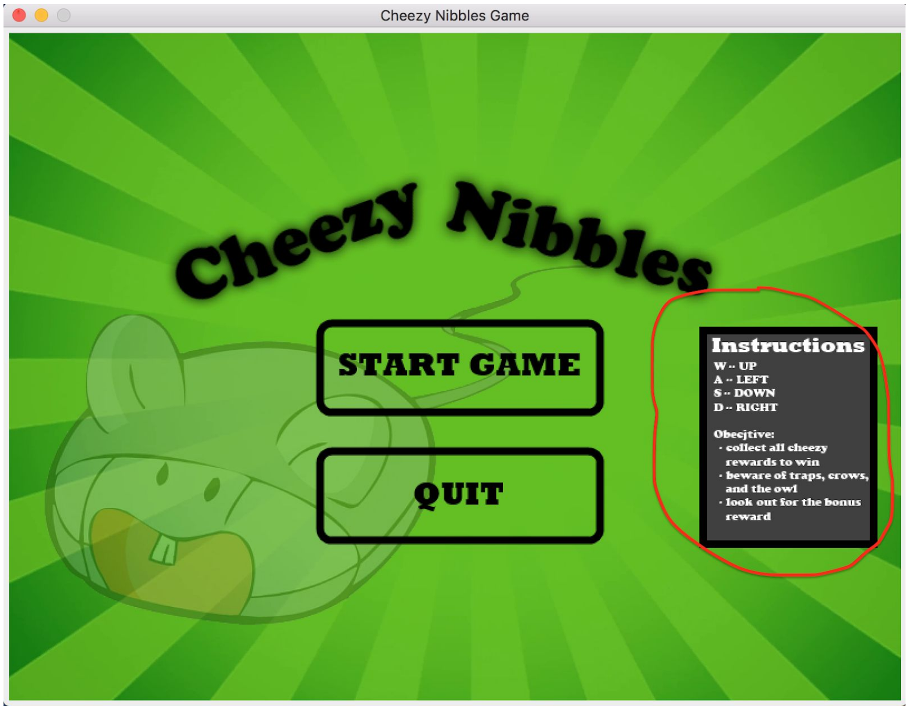
The build artifacts can be found in the artifacts folder. This folder includes the jar file for the game and the folder containing all the JavaDocs files.

Game Tutorial

Brief

The game's objective is to collect all the cheezy rewards that lay on the map while avoiding all enemies, especially the owl. The mouse needs to keep moving around or else the enemies will catch up and if they do, it's game over. As well as if your score goes to zero the game will also end.

Start Screen



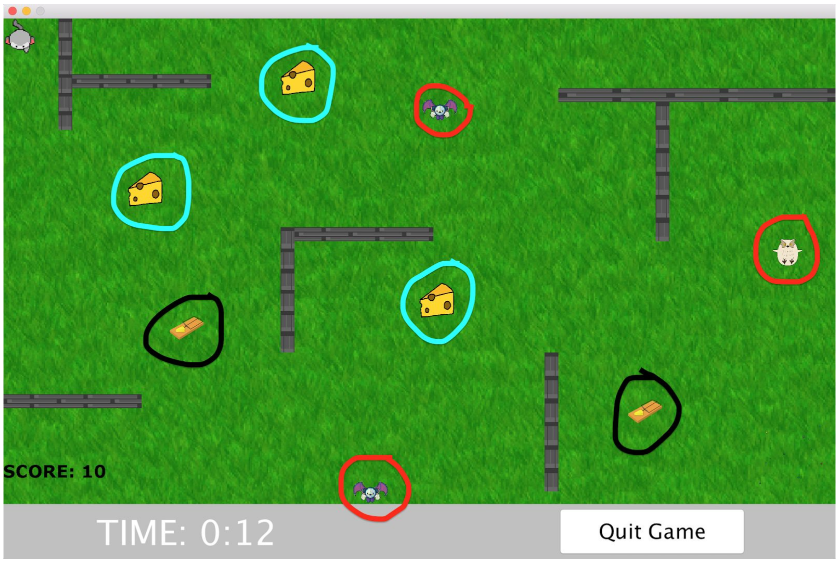
Instructions

W -- UP
A -- LEFT
S -- DOWN
D -- RIGHT

Obecjtive:
- collect all cheezy rewards to win
- beware of traps, crows, and the owl
- look out for the bonus reward

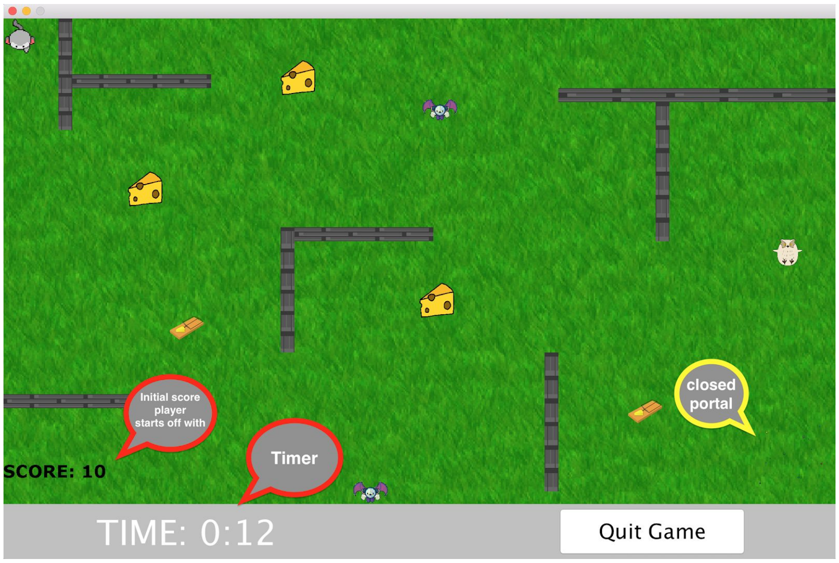
When the game launches, be sure to get a glance at the instructions for the game and understand the objectives as well. The WASD keys are for movements, but there are limits to where the player can move to. Click the start button once you are ready to play, or quit the game if you do not feel like playing. Once the game starts, the timer will start immediately, so make your way through the map and finish the game as quickly as you can.

Game Entities



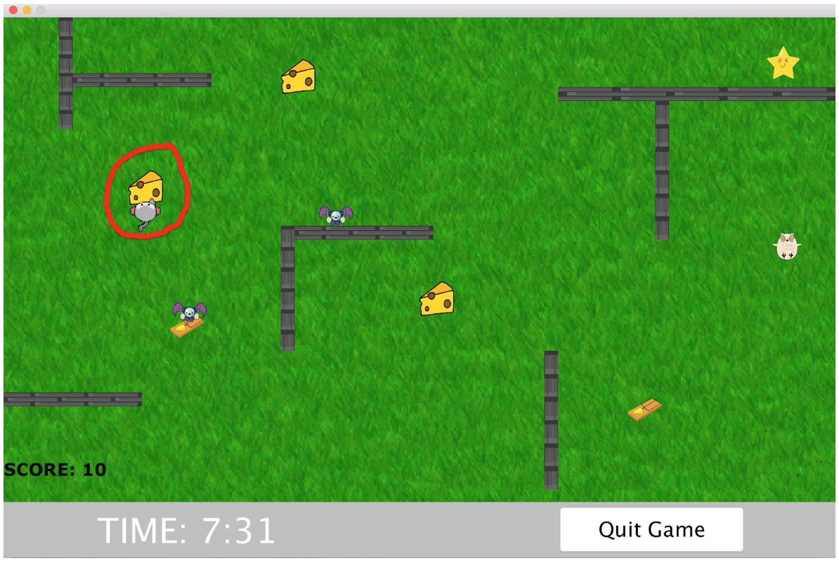
Upon entering the game, the player will see the cheezy rewards (blue) arranged across the board and among other entities as well. Surrounding the rewards are the enemies (red) and a few traps (black) that are laid around to increase the difficultness that lessen your score. On the other side near the positioning of the end portal lies the owl. This enemy is more aggressive and moves much faster than the other two. The only way any of the enemies will come and attack you is if you are within the range of their line of sight. The owl's line of sight is much larger than the other two enemies, hence makes it harder to hit the end zone. Once the player is hit by one of the enemies, its automatic game over and your score plummets to zero. Also, if you land on a trap, it will only decrease your score, but if the score hits zero, then it's game over as well.

Player Stats

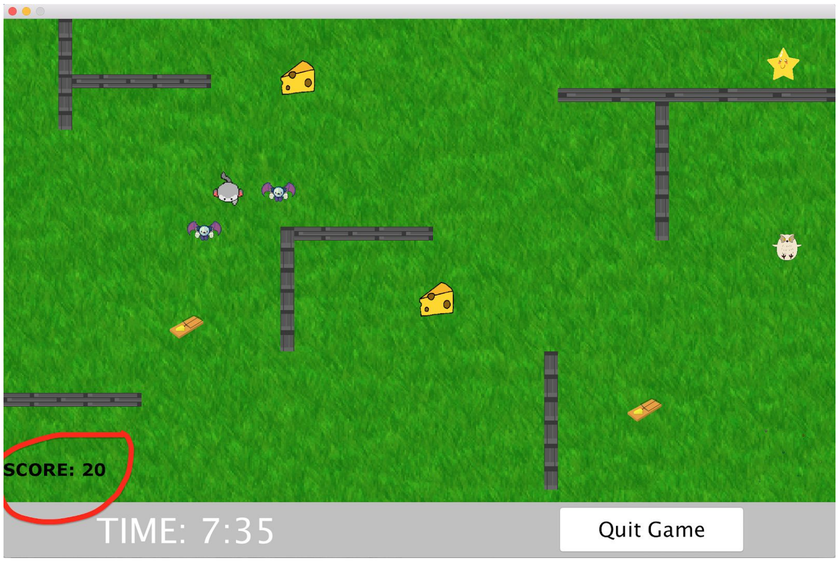


The player can also notice the stats when coming into the game, documenting their score and their time. The timer will be ticking until the game is over. Moreover, the portal remains closed as all the reward items have not been collected. The time and score for which the player finishes with will be displayed onto the end screen when they win or when they lose.

Regular Rewards

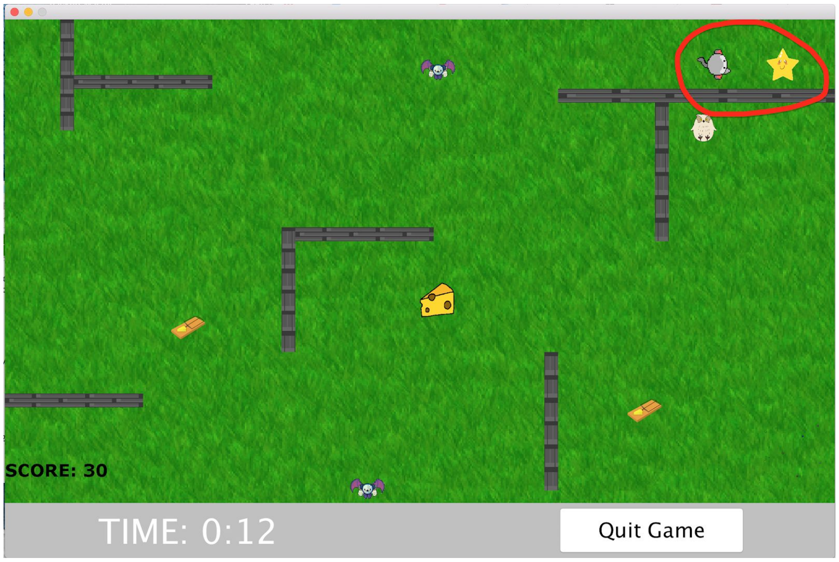


The game map has several rewards in the form of the player's favourite food choice to collect. The player's score is based on the gathering of these reward items on the map as it is essential for opening the portal and winning the game. Move the player across the map and compile all reward items.

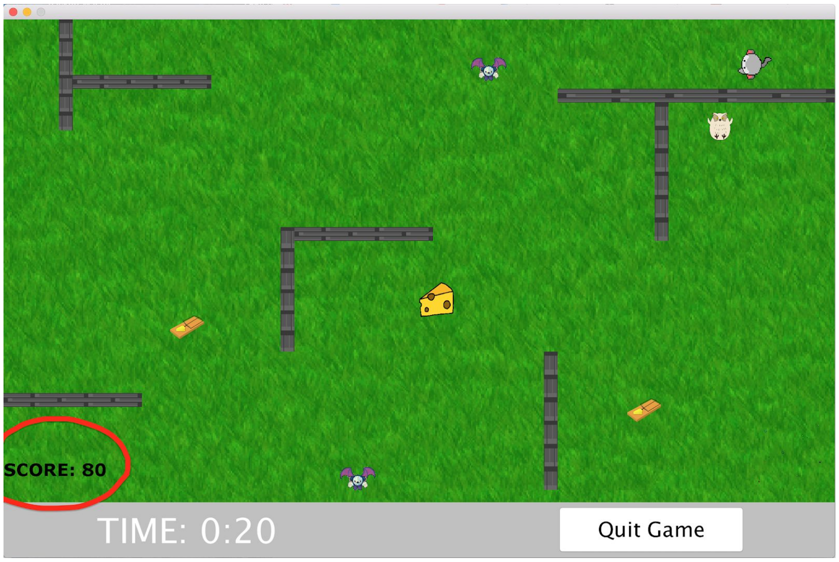


After collecting a reward, the player's score increases by 10 and that is the same for all rewards in the form of a cheese slice.

Bonus Reward

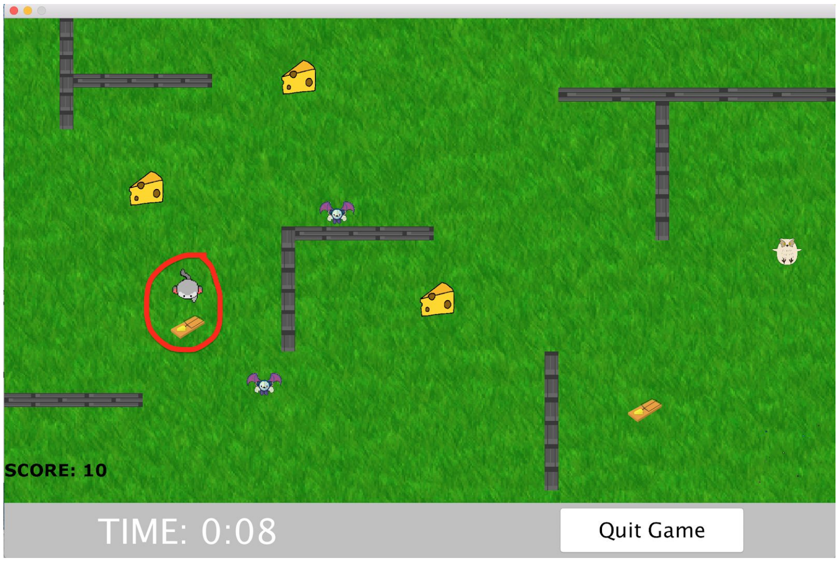


Besides the regular rewards that the player can collect, there is also an added bonus reward on the map that is randomly generated every game. The player can collect this reward once it appears on the map for a tremendous increase in score. This is strictly optional as it is for the player's own benefit to increase their own score, but not necessary to open the end portal.

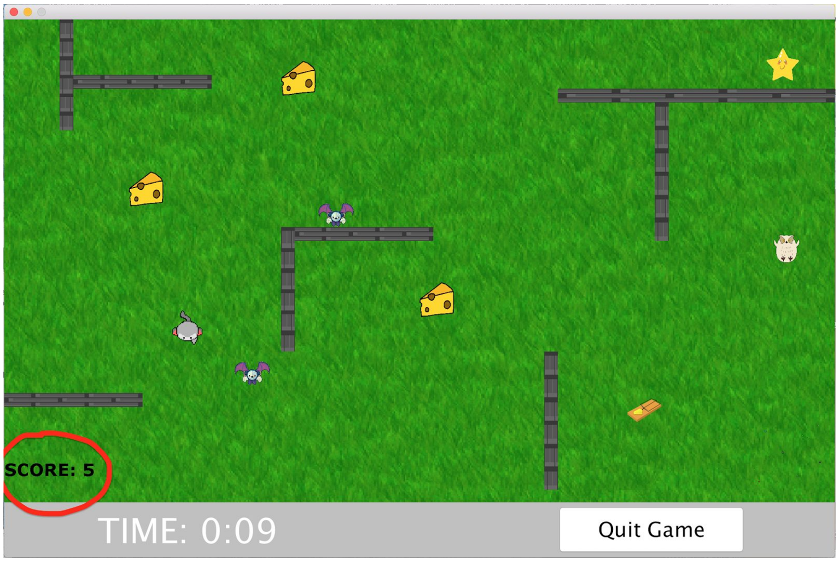


Subsequently, once the bonus reward is collected, it will disappear and the player's score will increase by 50.

Traps

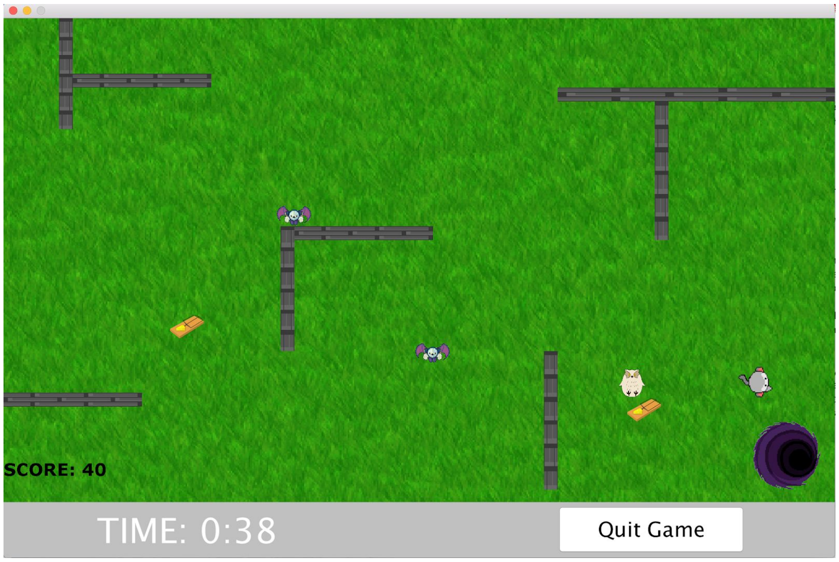


With a few traps on the map, the chances of landing on a trap is still probable. Once the player lands on a trap, the trap will perish and the score of the player will be deducted and if it hits zero, the game will be over.



If the player steered into a trap, the player's score will be reduced by 5. This rule applies to all the traps on the map.

Portal Opens



Once all the cheezy rewards have been collected from the map and the player is still alive, the win portal will open and awaits for the player to enter and conquer the game. This of course does not need the help of the bonus reward. The score and time that the player enters the portal with will be presented on the winning screen. Play again and see if you can beat your previous best time.