Final Project Report: Star Defender

Joshua Johnson and Nick Owens

Work Distribution

Both members of this team contributed equally to the final project. A chart displaying the tasks taken on by each member and their relative sizes (difficulty to complete) are listed below.

Task	Size of Task	Group Member
Death Star	Medium/Large	Joshua
Asteroids	Medium	Nick
Mini-Map	Medium/Large	Nick
Coordinate System	Medium	Joshua
Tie Fighter Tracking	Medium	Nick
Tie Fighter Lasers	Large	Joshua
Tie Fighter Explosions	Small	Joshua
Leaderboard	Large	Nick
User Ship Steering	Small/Medium	Joshua
User Ship Lasers	Medium	Joshua
Instruction Manual	Small	Nick
Game Design Document	Small	Joshua
Music	Small	Joshua
Main Menu	Medium	Nick
Home Button	Small/Medium	Nick

Star Defender: Instruction Manual

Start New Game: If the user is currently at the main menu screen, they must press “START GAME” to advance. A difficulty (easy or hard) must be selected at the next screen before the game begins.

Exit Game: If the user wishes to exit the application, they must return to the main menu screen and press the “QUIT GAME” button.

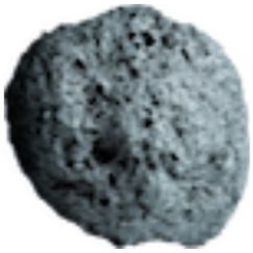
Home Button: If at any time the user wishes to return to the main menu, whether than be from the difficulty screen or their current game, they can press the “HOME” button at the top right of the window. From there, a new game can be started or the user can exit the program.

Game Controls:

“w” or “up” = move ship up
“a” or “left” = move ship left
“s” or “down” = move ship down
“d” or “right” = move ship right “space” = shoot laser

Game Objects:
User Ship —>

Asteroid —>



Enemy Ship —>

Death Star —>

Game Rules: The main objective of the game is to destroy the Death Star which remains static in relation to the game map. The user begins with three (3) lives and must avoid asteroids and enemy Tie Fighters along the way. The asteroids are static objects and will destroy the user ship (losing 1 life) if a collision occurs, while the enemy ships actively pursue the user and can destroy their ship (losing 1 life) if a collision occurs or their lasers hit the user ship. The user can combat the Tie Fighters by shooting them with their lasers, destroying one with a single blast. The user can also destroy asteroids with a single blast. In order to destroy the Death Star, the user must fire lasers to hit the target until the health bar is fully depleted. Until then, enemy ships will re-spawn from the Death Star and from the left and right sides of the map with a delay in order to maintain the maximum number of ships, which fluctuates with difficulty.

Differences In Difficulties: There are several differences between the two difficulties, easy and hard. Hard mode has a higher maximum number of enemy ships, as mentioned above. Furthermore, enemy ships will respawn at a faster rate and fire lasers at the user ship. Also, there will be more asteroids throughout the map and the Death Star will require more hits to destroy.

Post Game: Upon destruction of the Death Star, the game will end and the user's "score" is reflected by the number of remaining lives. The user can then input their name and an entry will appear in the leaderboard following the game, ranking them in order of lives left, while each subset is sorted in order of entry. If the user fails to destroy the Death Star before losing all of their lives, they can press the "HOME" button to return to the main menu and play again or exit the application.

Difficulties Encountered

For the most part, the development process of Star Defender was relatively smooth and absent of any major obstacles, although there were a few small hiccups along the way. For instance, the leaderboard is updated and subsequently stored using the loadStrings() and saveStrings() functions provided in Processing; the root of the issue was a discrepancy in the location the files were loaded from and saved to. The function loadStrings() pulled from a file specified within the "Data" folder inside the project folder while saveStrings() only saved the file within the project folder. Therefore, when the Leaderboard class tried to load, update, and save the new leaderboard, the new changes were made but in a separate document. When the next load was called, loadStrings() pulled from the original leaderboard instead of the the updated one, creating an array out of bound error. Although this may seem minute, I didn't realize until repeatedly going over the code and trying any fix I could think of. Eventually, I did a quick search online about the functions' behavior and found a solution: encapsulate the name of the file you wish to save within the dataPath() function, storing the file at its current location in the directory, i.e. the "Data" folder.

Furthermore, implementing the mini-map was slightly challenging in that its behavior greatly contrasted that of our other game objects. In the game, the Millennium Falcon always remains in the middle of the screen and the remaining game objects move in relation to the user ship, while the mini-map requires everything to move except the Death Star. This complicated the display of enemy ships and the user ship within the mini-map, but after taking a step back, the adjustments were pretty intuitive and only required an understanding of relational movement and how that intertwined with the map() function provided in Processing.

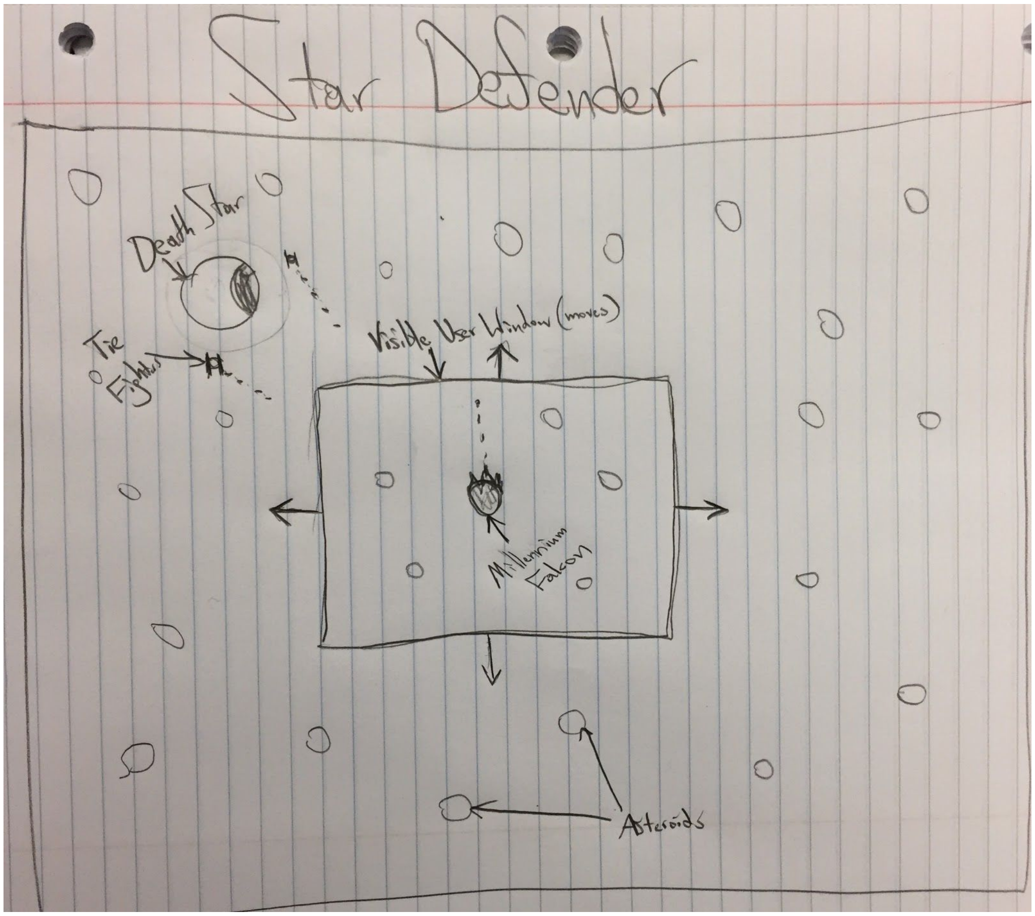
Final Project: Game Design Document

Game Title: Star Defender

Description: Star Defender is a 2D, arcade-style shooting game. The user controls a spaceship (the Millenium Falcon) which navigates space, evading obstacles and trying to destroy its target (the Death Star). Obstacles include stationary asteroids as well as enemy spaceships (Tie Fighters) which shoot at the user's spaceship. The user steers their spaceship using W,A,S,D keys or up, down, left, right arrow keys on the keyboard. The space bar will control shooting which can be used to shoot enemy spaceships or the Death Star.

Goal: The user wins the game if they destroy the Death Star before losing their three lives.

Navigation:



User navigation will be similar to the popular game, Bosconian. The user spaceship will remain in the center of the visible game window. As the user moves around the map, the game window updates to bring into view game objects that are present in that area of the map.

Game Components:

I. Main Menu
    A. Option to start game
        1. Easy Level
        2. Hard Level
            a) More enemy spaceships
            b) Enemy spaceships move faster
            c) More asteroids
    B. Option to exit
    C. User Input: the user clicks on buttons using their mouse

II. User Spaceship (Millenium Falcon)
    A. Use pixelated image of millenium falcon to represent the user spaceship
    B. Will always remain in the center of the visible game window
    C. Will rotate based on direction of movement determined by the W,A,S,D keys or up, down, left, right arrow keys
    D. Will shoot "lasers" in direction of the user's movement when the space bar is pressed
    E. Will explode if hit by an asteroid, the Death Star, an enemy ship, or an enemy laser (one hit to kill)

III. Asteroids
    A. Simple circles drawn using Processing shapes
    B. Can be destroyed by user ship lasers

IV. Enemy Spaceships (Tie Fighters)
    A. Will be drawn using simple Processing shapes
    B. Will spawn from the Death Star's location
        1. Time delay for spawning determined by difficulty level
        2. Number of enemy ships determined by difficulty level
    C. Will explode if hit by laser (one hit to kill)
    D. Will chase after the user spaceship
    E. Will shoot lasers at user spaceship

V. Target to Destroy (Death Star)
    A. Death Star will explode if hit by laser enough times (TBD, but roughly 20 hits to kill)
    B. Health bar will appear over the Death Star indicating the damage it has taken

VI. Mini-Map
    A. Resides in corner of user's screen at all times
    B. Displays locations of user spaceship and the Death Star in relation to the map
    C. Displays location of enemy ships, but not asteroids

VII. In-Game Lives Display

   A. Small images of the Millenium Falcon will show the user how many lives they have remaining (3 lives at game start)
   B. Every time the user spaceship explodes (due to collision with an asteroid, the Death Star, an enemy ship, or an enemy laser) the user will lose a life

VIII. Final Score Screen

   A. User's score is displayed to them
   B. Option to return to the main menu
   C. Option to exit

Possible Game Improvements: (to be added if time permits)

I. High Score Screen

   A. Provide users with the option to save their name and score on the final score screen
   B. All users would see the updated high scores list at then end of their game

II. In-Game Boosts

   A. If user reaches a boost icon in the map, they get a special power (usually for a limited duration of time)
   B. Boost options
      1. Rapid fire shooting (10 seconds)
      2. Invincibility (10 seconds)
      3. Extra life
      4. Double points (20 seconds) (depending on scoring system)
      5. Double user speed (20 seconds)

III. Variety of Enemy Spaceships

   A. Create new enemy ships that are more complex

IV. Multiple game maps/rounds