**Major Programming Project Package**

**Github Link:** [**https://github.com/weihanli101/Cygnus-Strike**](https://github.com/weihanli101/Cygnus-Strike)

**Program Overview/Introduction**

A basic 2-D space themed shooter, where the player will move the Spaceship via finger.

* **Menu**: user will be directed to main menu at the start of the game and the menu will display the play, profile, option, help, shop, and exit buttons
* **Controllable Unit**: Spaceship that is controlled by finger movement and shoots projectiles when the “shoot button” is pressed
* **Background**: A starry background that scrolls vertically downwards to simulate movement from the Spaceship
* **Obstacles:** enemy space ships will spawn and fly towards the user’s spaceship (interaction with enemy space ships is possible)
* **Items**: Power ups will spawn and fly towards the user’s spaceship (shooting down a power up gives a time bonus)
* **Score:** Accumulated throughout the level by surviving, destroying enemy space ships and a final score will be displayed on the game over screen
* **Goal:** user is trying to accumulate the highest score possible in the time given
* **Game Over Condition:** running out of time or losing all of your stock

**Disclaimer/List of Known Bugs**

* space ship “teleports”
* shop button exits the app for some reason
* menu music doesn’t stop when loading play screen if another screen was visited previously
* sometimes aliens spawn on border of screen
* occasional “LSD” mode
* the hit-boxes are slightly off
* scaling issue which causes the shoot button to not be displayed on some resolutions
* app crashes after a certain number of retries (may be caused by not disposing of previous screens?)

**User Manual**

|  |  |
| --- | --- |
|  | 1)Play Button  2)Profile Button  3)Option Button  4)Help Button  5)Shop Button  6)Exit Button |
|  | 7) Stock  8) Time  9) Current Score  10) Enemy Spaceships (Aliens)  11) Power Up  12) User’s spaceship  13) Shoot Button  14) Laser Ball |
|  | 15) Final Score  16) Retry button  17) Exit button |

In Depth Walkthrough of all Aspects of Cygnus Strike

When Cygnus Strike starts up, the user will be directed to the main menu screen. Within this screen there are 6 buttons that each perform a function when the user interacts with them. The profile(2), option(3), help(4), and shop button(5) all navigate to a contentless screen containing a back button, while the exit button(6) exits the app and returns the user to their home screen. The play button(1) is currently the only button that will redirect the user to a part of the game with actual content. So, when the play button is pressed, it will load the gameplay aspect of Cygnus Strike.

The gameplay is made up of several components. First, the top section of the game screen contains information pertaining to stock(7), time(8), and score(9). Stock, in other words the number of lives that the user has, starts at a value of 3 and is decremented by 1 everytime the user is hit by an enemy space ship (alien). When the stock hits 0, the user achieves a “game over” and is redirected to the game over screen, which will be discussed later. The time in this game is a counter that decrements every second and also when an alien manages to reach past the bottom of the screen. The time is incremented through power ups, which will also be discussed later. When the time hits 0. the user again receives a “game over” and is redirected to the game over screen. The user’s score is incremented by killing aliens and surviving for a set amount of time (based on seconds passed). The score essentially determines how good the user is at the game.

Secondly, objects that interact with the user’s spaceship. Currently, there are two objects that move towards the user and affect the gameplay: the red, enemy alien(10) and the flashing power up(11). The alien is something that spawns every few seconds at a random position off the screen. The alien flies towards the user at a randomized speed and it is the user’s goal to either shoot down the alien or dodge it. Doing either action will affect how the game responds, as mentioned previously. The power up is an object that spawns at set intervals, at a random position off screen. It flies towards the user at a set speed and only when the user shoots it down will the power up (which is a time bonus) be obtained.

Thirdly, the user’s spaceship(12). There are two main aspects at this point in time that allow for interaction with the spaceship. One of the them being the spaceship’s movement that is controlled by the user’s touch, which means that the spaceship will follow wherever the user’s finger is. The other being the shoot button(13), which allows the spaceship to fire an indefinite amount of laser balls(14). These laser balls have the ability to remove enemy aliens without penalty, as well as obtain power ups. The plan at a later date is to add in the option for movement controlled by tilt, as well as an ammo system for the bullets.

When a “game over” is achieved, the user will be directed to the game over screen. This screen contains a final score(15), a retry button(16), and an exit button(17). The final score tells the user what their score was right before their loss, the retry button allows the user to enter the gameplay for another round, and the exit button redirects the user to the main menu screen.

**Scratch Program Descriptions**

|  |  |
| --- | --- |
| **Branch Name** | **Description** |
| ScratchAlienArray\_noRect  <https://github.com/weihanli101/Cygnus-Strike/tree/ScratchAlienArray_noRect> | putting objects into separate classes & moving sprites without usage of rectangles |
| ScratchBackground  <https://github.com/weihanli101/Cygnus-Strike/tree/ScratchBackground> | displaying an image using a batch and adding a scroll effect to it |
| ScratchBox2D  <https://github.com/weihanli101/Cygnus-Strike/tree/ScratchBox2D> | learning Box2D basics |
| ScratchButton  <https://github.com/weihanli101/Cygnus-Strike/tree/ScratchButton> | tested out the concept of using buttons with a custom image and using a texture packer to pack the different button states into one file |
| ScratchConstructors\_Objects  <https://github.com/weihanli101/Cygnus-Strike/tree/ScratchConstructors_Objects> | learning the proper usage of objects |
| ScratchMovingFinger  <https://github.com/weihanli101/Cygnus-Strike/tree/ScratchMovingFinger> | moving an image of a spaceship using the touch input from a finger |
| ScratchScreenChange  <https://github.com/weihanli101/Cygnus-Strike/tree/ScratchScreenChange> | how to change between 2 different screens |

**List of Sources**

|  |  |
| --- | --- |
| **Location** | **Sources** |
| Our Github | <https://github.com/weihanli101/Cygnus-Strike> |
| Alien.java | <https://github.com/libgdx/libgdx/wiki/A-simple-game>  <http://ics3ui.sgrondin.ca/ss14/ArrayLists.html>  Grondin |
| Background.java | <https://code.google.com/p/libgdx-users/wiki/ScrollingTexture> |
| Camera.java | <https://github.com/libgdx/libgdx/wiki/A-simple-game> |
| GameOverScreen.java | <https://code.google.com/p/libgdx-users/wiki/ScreenAndGameClasses> |
| HelpScreen.java | <https://code.google.com/p/libgdx-users/wiki/ScreenAndGameClasses> |
| Laser.java | <https://github.com/libgdx/libgdx/wiki/A-simple-game>  Matt  Grondin |
| MainMenuScreen.java | <https://code.google.com/p/libgdx-users/wiki/ScreenAndGameClasses> |
| MyGame.java | <https://code.google.com/p/libgdx-users/wiki/ScreenAndGameClasses> |
| OptionScreen.java | <https://code.google.com/p/libgdx-users/wiki/ScreenAndGameClasses> |
| PlayScreen.java | <https://code.google.com/p/libgdx-users/wiki/ScreenAndGameClasses> |
| Powerups.java | <https://github.com/libgdx/libgdx/wiki/A-simple-game>  <https://github.com/libgdx/libgdx/wiki/2D-Animation>  Grondin |
| ProfileScreen.java | <https://code.google.com/p/libgdx-users/wiki/ScreenAndGameClasses> |
| ShopScreen.java | <https://code.google.com/p/libgdx-users/wiki/ScreenAndGameClasses> |
| Spaceship.java | <https://github.com/libgdx/libgdx/wiki/A-simple-game> |
| “ScratchAlienarray\_noRect” | Matt Brock |
| “ScratchBackground” | <http://gamedev.stackexchange.com/questions/71198/how-do-i-make-a-background-fill-the-whole-screen-in-libgdx>  <https://code.google.com/p/libgdx-users/wiki/ScrollingTexture> |
| “ScratchBox2D” | Jose Rivas  <http://www.gamefromscratch.com/post/2014/09/10/LibGDX-Tutorial-13-Physics-with-Box2D-Part-2-Force-Impulses-and-Torque.aspx>  <https://github.com/libgdx/libgdx/wiki/box2d#sprites-and-bodies> |
| “ScratchButton” | Matt Brock  <http://stackoverflow.com/questions/21488311/libgdx-how-to-create-a-button>  <http://gamedev.stackexchange.com/questions/60123/registering-inputlistener-in-libgdx> |
| “ScratchScreenChange” | <https://code.google.com/p/libgdx-users/wiki/ScreenAndGameClasses> |

**Notes to Future Programmers**

* learn how to use github properly before diving in, ex. learn: terms, structure and functions etc.
* don’t let restructuring your code **multiple** times get to you (it’s worth it)
* don’t be afraid to ask your peers for help

**Lesson of Favourite Programming Technique**

The lesson that was not discussed in class and that also proved to be one of the most important to our project’s gameplay is arraylists. Arraylists were incorporated into nearly every object in our game (aliens, powerups, lasers). We populated the arraylists (AL) by adding an object based on a timer. The now populated AL was then referenced in the render method and cycled through using an iterator and looped using a while loop. Within the loop we check for hit collision with other objects, increment score, decrement health and removes the sprite if needed.