CO452 PROGRAMMING CONCEPTS
PR1 GROUP PROJECT

GAME TITLE: METAUNIVERSE

AUTHORS: **Name: Marius Boncica Bucks New ID: 22045429**

Name: Martin Konecky Buck New ID: 22045464

GitHub Repository: CW2-CO452/Game1 at main · UnicoAI/CW2-CO452 (github.com)

GitHub WIKI: METAUNIVERSE GAME CO452 CW2 · UnicoAI/CW2-CO452 Wiki (github.com)

Video Metauniverse Game Playing: https://youtu.be/RnkUMHaQTeg

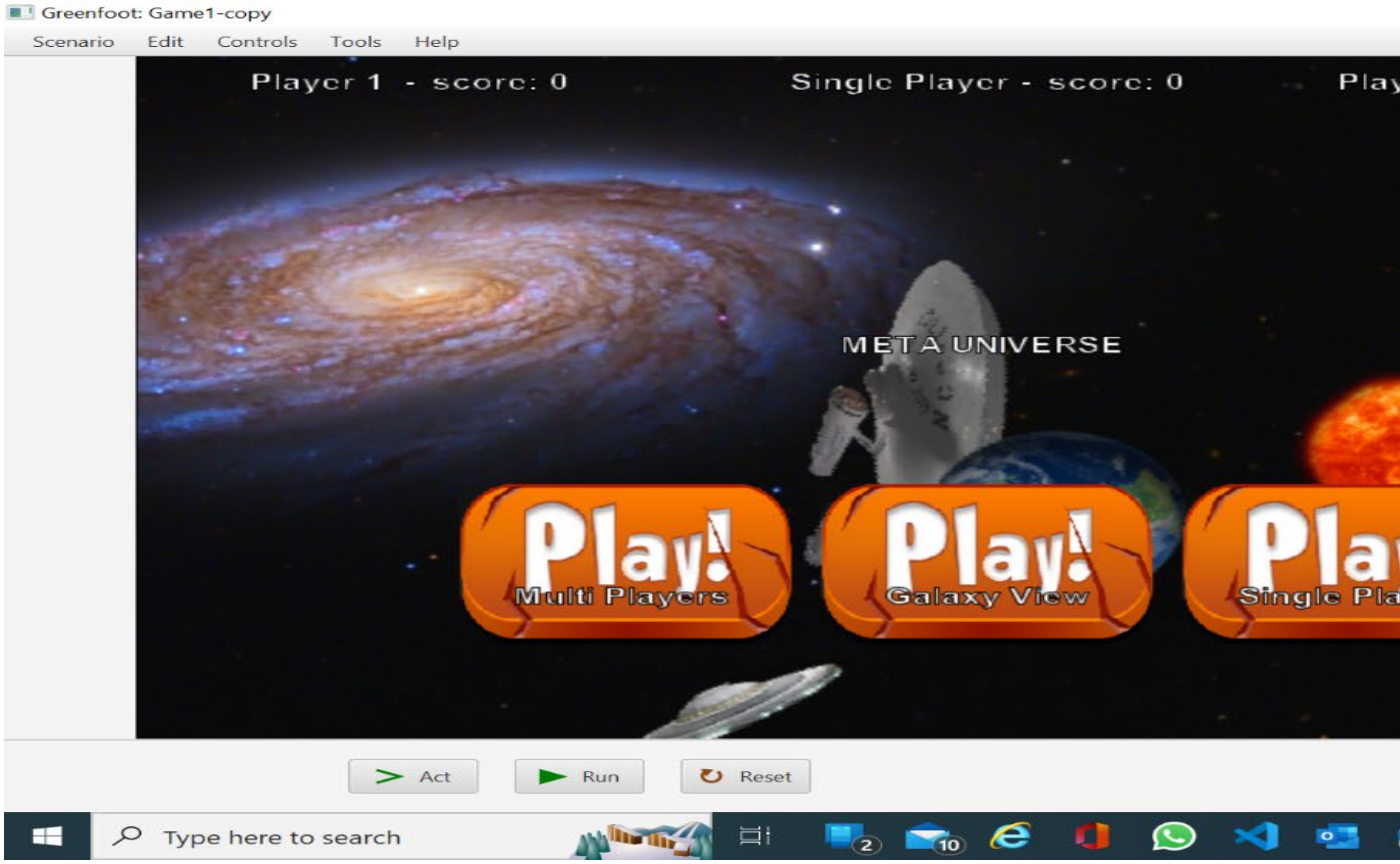
WEB PRESENTATION

https://sway.office.com/b0bCY49GeHyZSwiB?ref=Link

CO452 PROGRAMMING CONCEPTS

AUTHORS:

Go to this Sway



CONTENTS

GAME TITLE

METAUNIVERSE

Contributors: ***Marius Boncica and Martin Konecky***

PR1 Group Project – CO452 Programming Concepts

Video Metauniverse Game Playing Latest Version

https://youtu.be/RnkUMHaQTeg

INTRODUCTION

METAUNIVERSE Game is about protecting the earth from the aliens and approaching steroids.

It is a 2D green foot game, but we animated the objects to approach a 3D visual aspect.

Game contains three levels:

1METAUNIVERSE GALAXY VIEW:

Player will navigate space to shoot aliens.

When in touch with Sun effects are created and text alert pop up to press Spacebar.

The game is limited to 1 minute playing.

It is a good start to accommodate user for the next level use of control commands.

2 METAUNIVERSE SINGLE PLAYER:

User will need to shoot aliens and collect boosters to increase movement speed.

A certain number of points are necessary to destroy the approaching asteroid.

Animation effects starts when touching asteroid.

Text Alert will pop up if player reach enough power to destroy the asteroid.

Text Alert Actions Executed

We use Simple timer to limit the time playing to 1 minute.

3 METAUNIVERSE MULTIPLAYER

User will need to shoot aliens and collect boosters to increase movement speed.

A certain number of points are necessary to destroy the approaching asteroid.

Text Alert will pop up if player reach enough power to destroy the asteroid.

Text Alert Actions Executed

We use Simple timer to limit the time playing to 1 minute.

DESIGN CONSIDERATION

MAPPING

At the start at the game, we design a map of the space and being a group project, we decide to contain all the objects created to design the map into an array and create a map using index of array to add objects on screen map.

We start collecting images and create new animations in photoshop.

We used of the import of the Gif Image class to scale the gif images and used them frame by frame.

STRUCTURE

The game contains **7 Worlds classes**:

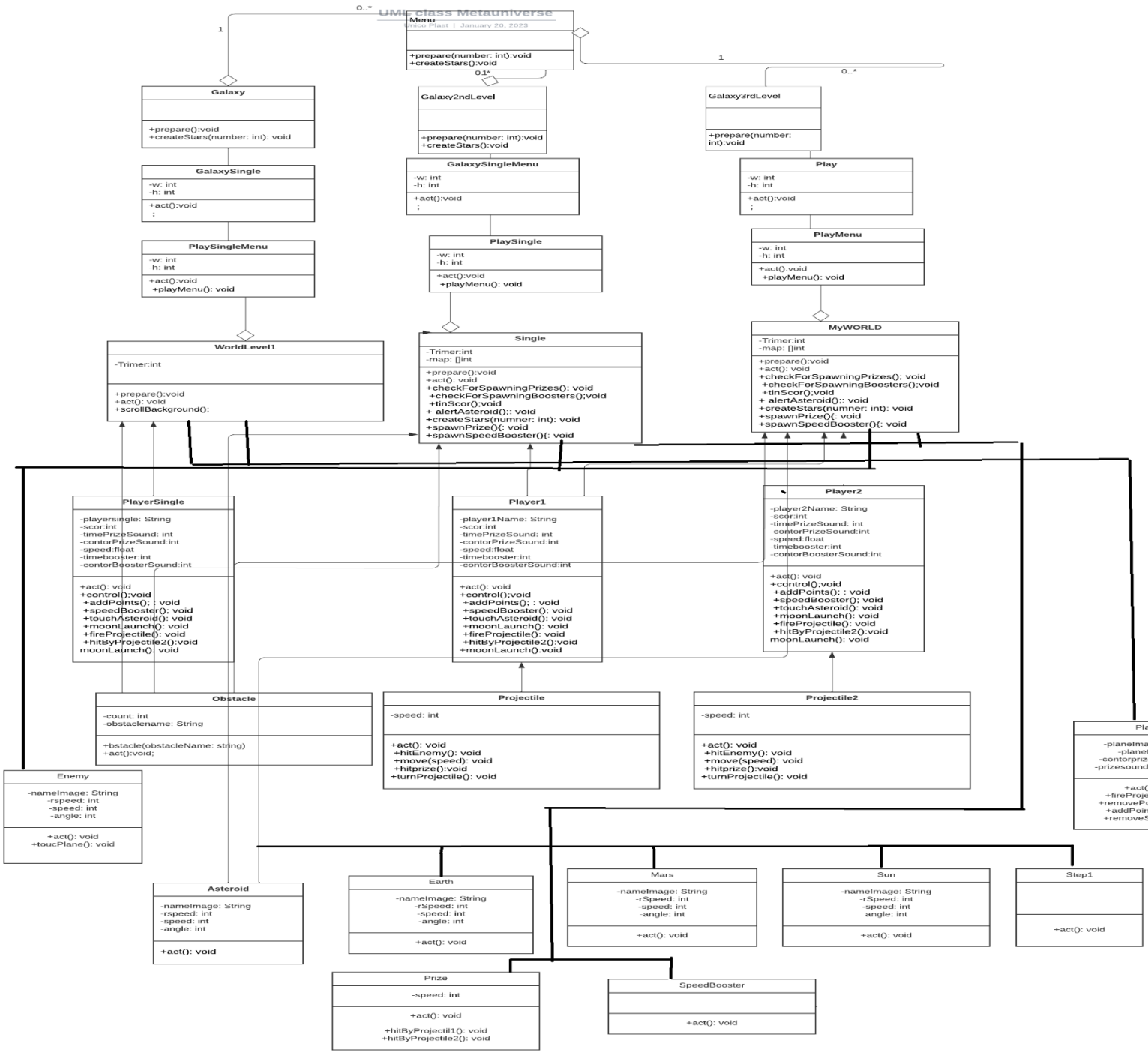
Menu class was created for start game menu and contains button (objects) that redirect on click event user to other worlds classes as:

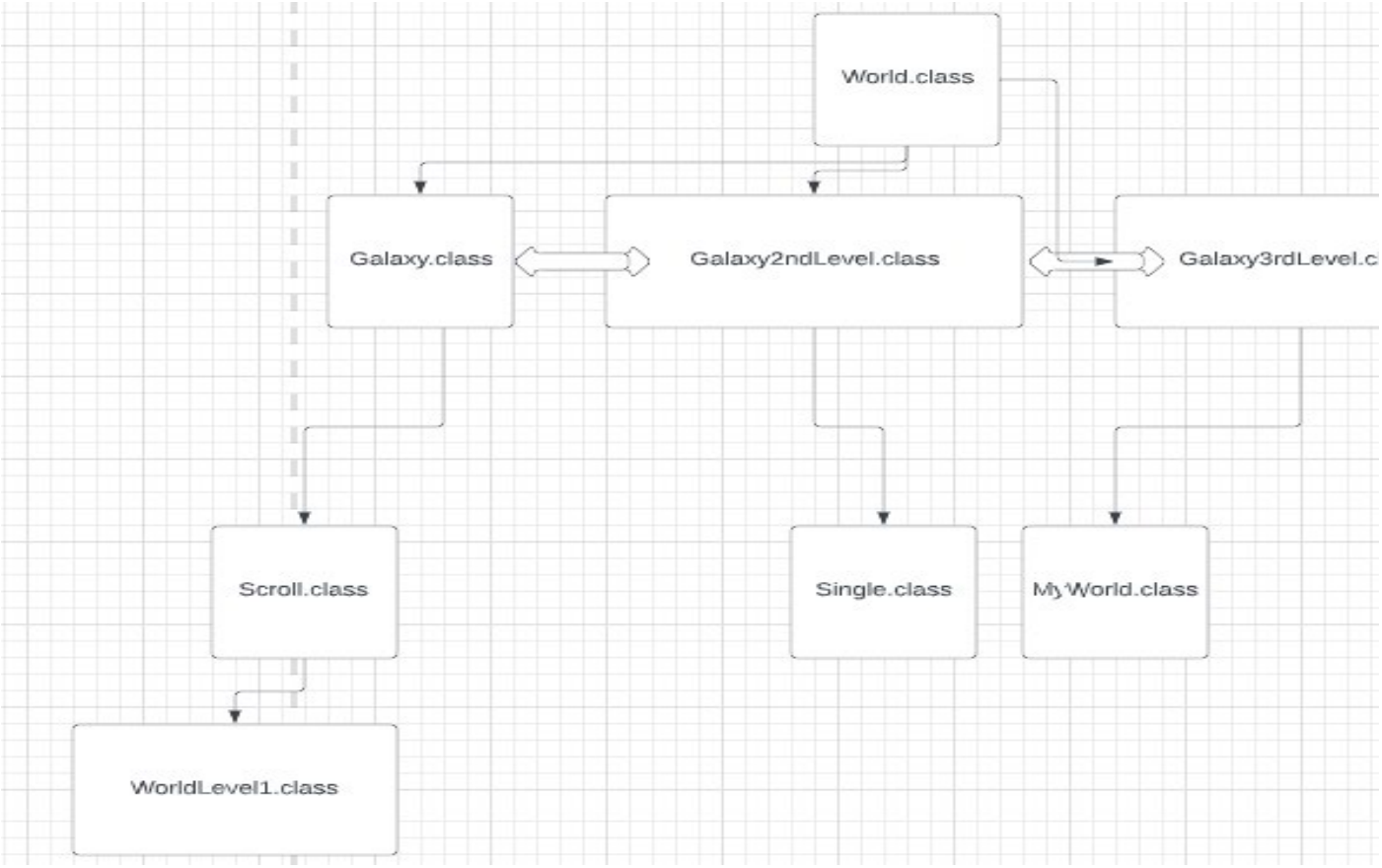
Galaxy Class—contains instruction and button to redirect to WorldLevel1 Class which is a single player galaxy navigation and aliens shooting game

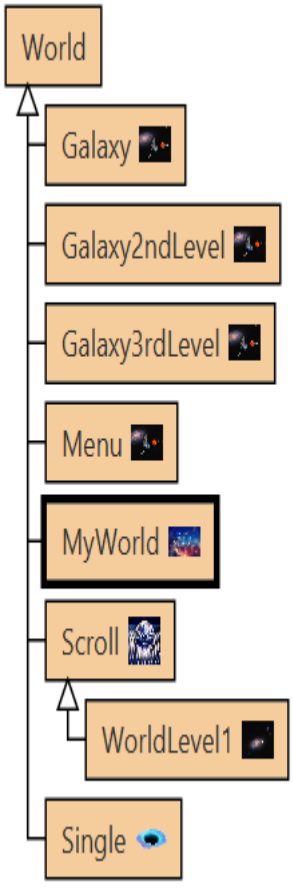
Galaxy2ndLevel Class-- contains instruction and button to redirect to Single Class which is a single player Asteroid removal target game.

Galaxy3rdLevel Class-- contains instruction and button to redirect to World Class which is a multiplayer player Asteroid removal target game.

UML CLASS DIAGRAM

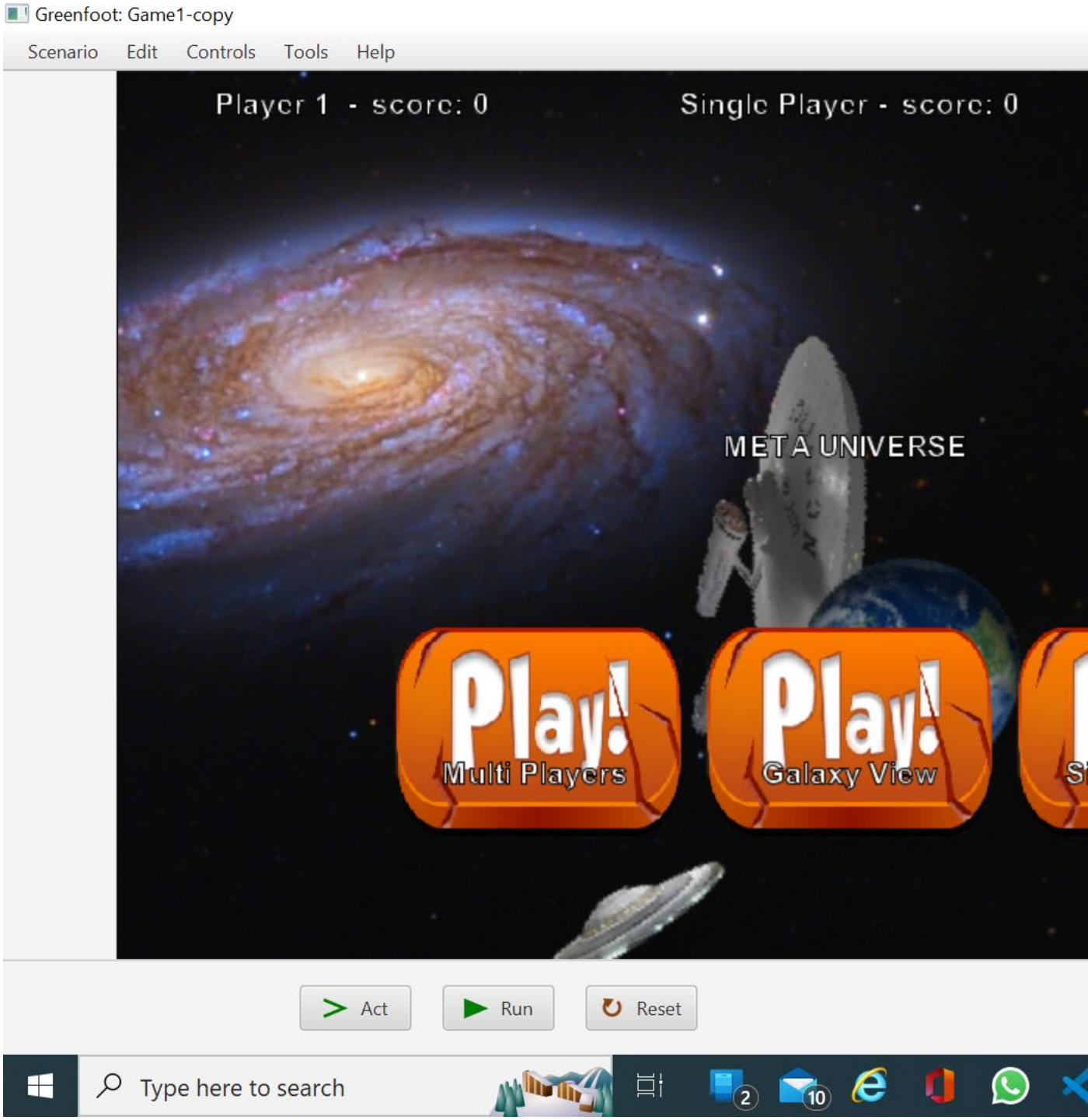

UML class Metauniverse
Unico Plast | January 20, 2023
Menu
+prepare(number: int):void
+createStars():void
0..*
1
0..1*
1
0..*
Galaxy
+prepare():void
+createStars(number: int): void
Galaxy2ndLevel
+prepare(number: int):void
+createStars():void
Galaxy3rdLevel
+prepare(number: int):void
GalaxySingle
-w: int
-h: int
+act():void
;
GalaxySingleMenu
-w: int
-h: int
+act():void
;
Play
-w: int
-h: int
+act():void
;
PlaySingleMenu
-w: int
-h: int
+act():void
+playMenu(): void
PlaySingle
-w: int
-h: int
+act():void
+playMenu(): void
PlayMenu
-w: int
-h: int
+act():void
+playMenu(): void
WorldLevel1
-Trimer:int
+prepare():void
+act(): void
+scrollBackground();
Single
-Trimer:int
-map: []int
+prepare():void
+act(): void
+checkForSpawningPrizes(); void
+checkForSpawningBoosters();void
+tinScor():void
+ alertAsteroid():: void
+createStars(numner: int): void
+spawnPrize(){ void
+spawnSpeedBooster(){: void
MyWORLD
-Trimer:int
-map: []int
+prepare():void
+act(): void
+checkForSpawningPrizes(); void
+checkForSpawningBoosters();void
+tinScor():void
+ alertAsteroid():: void
+createStars(numner: int): void
+spawnPrize(){ void
+spawnSpeedBooster(){: void
PlayerSingle
-playersingle: String
-scor:int
-timePrizeSound: int
-contorPrizeSound:int
-speed:float
-timebooster:int
-contorBoosterSound:int
+act(): void
+control():void
+addPoints(): : void
+speedBooster(); void
+touchAsteroid(): void
+moonLaunch(): void
+fireProjectile(): void
+hitByProjectile2():void
moonLaunch(): void
Player1
-player1Name: String
-scor:int
-timePrizeSound: int
-contorPrizeSound:int
-speed:float
-timebooster:int
-contorBoosterSound:int
+act(): void
+control();void
+addPoints(): : void
+speedBooster(); void
+touchAsteroid(): void
+moonLaunch(): void
+fireProjectile(): void
+hitByProjectile2():void
+moonLaunch():void
Player2
-player2Name: String
-scor:int
-timePrizeSound: int
-contorPrizeSound:int
-speed:float
-timebooster:int
-contorBoosterSound:int
+act(): void
+cntrol():void
+addPoints(): : void
+speedBooster(); void
+touchAsteroid(): void
+mjoonLaunch(): void
+fireProjectile(): void
+hitByProjectile2():void
moonLaunch(): void
Obstacle
-count: int
-obstaclename: String
+bstacle(obstacleName: string)
+act():void;
Projectile
-speed: int
+act(): void
+hitEnemy(): void
+move(speed): void
+hitprize():void
+turnProjectile(): void
Projectile2
-speed: int
+act(): void
+hitEnemy(): void
+move(speed): void
+hitprize():void
+turnProjectile(): void
Enemy
-nameImage: String
-rspeed: int
-speed: int
-angle: int
+act(): void
+toucPlane(): void
Asteroid
-nameImage: String
-rspeed: int
-speed: int
-angle: int
+act(): void
Earth
-nameImage: String
-rSpeed: int
-speed: int
-angle: int
+act(): void
Mars
-nameImage: String
-rSpeed: int
-speed: int
angle: int
+act(): void
Sun
-nameImage: String
-rSpeed: int
-speed: int
angle: int
+act(): void
Step1
+act(): void
Prize
-speed: int
+act(): void
+hitByProjectil1(): void
+hitByProjectile2(): void
SpeedBooster
+act(): void


World.class
Galaxy.class
Galaxy2ndLevel.class
Galaxy3rdLevel.c
Scroll.class
Single.class
MyWorld.class
WorldLevel1.class


World
Galaxy
Galaxy2ndLevel
Galaxy3rdLevel
Menu
MyWorld
Scroll
WorldLevel1
Single

ACTORS

After creating the map, we create the actors as players, enemies and objects floating.



PLAYERS CLASSES

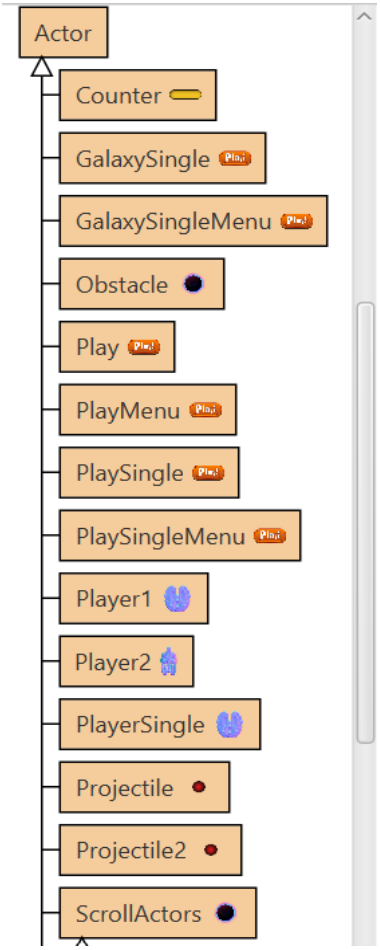
Player1 class

Player2 class

Player Single class

Plane class extends Scroll Actors class.

All players classes contain Constructor methods to scale image, methods to shoot (by creating objects on click event or key pressed event), removal of other objects if are touching and methods to increase/ decrease score if are touching specific objects.



ENEMIES AND OTHER OBJECTS

Enemy class – contains methods to remove object if is touching instances of projectile. Class or if are touching player. Class

Projectile classes that contain methods to remove at edge or when touching enemies.

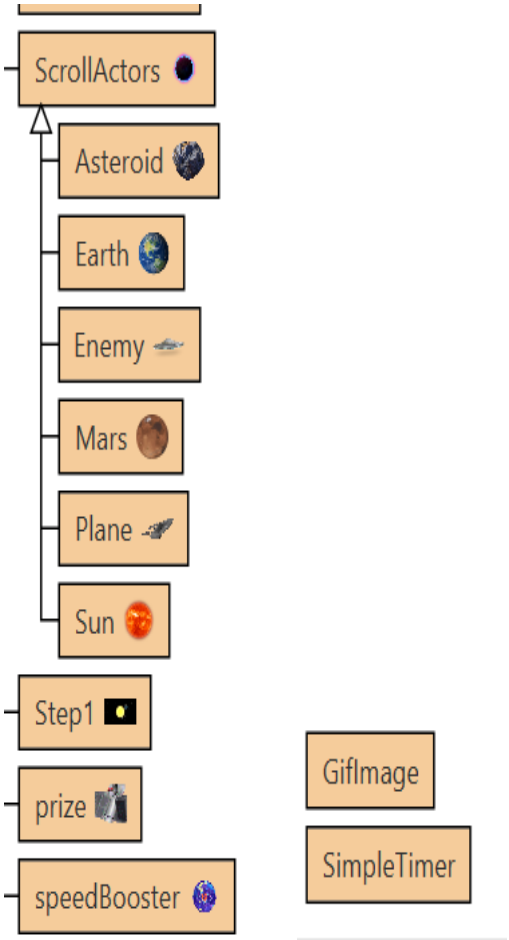
Sun, Earth, Asteroid, Step1 classes are created to populate the map and touching behaviours with player classes or projectile classes are inserted

Prize, speed Booster classes were created to give opportunity to the player to collect points or to increase speed if are touching or shooting them.

COUNTER AND SPEED

We use counter class to keep record of the points earn and we create text speed record menu for each player.

They are updating using methods described in players classes.



DEVELOPMENT OF THE GAME AND PROBLEM SOLVING

During the development and evaluating the game processes we encounter some issues with the Gif Images creating a java heap memory space error.

Problem was solved by using PNG images and create a method to turn and rotate and speed the image and achieve the look alike 3D model.

We add extra functionality to the menu by creating additional classes for each level that will inform user about instructions and control keys.

DEMONSTRATION OF GAMEPLAY AND TESTING

VIDEO METAUNIVERSE

We record our screen while playing all three levels of the game and the demonstration can be viewed at this link:

Video Metauniverse Game Playing:

https://screencast-o-matic.com/watch/c0Vq2kVwKKW

Video METAUNIVERSE Game Playing Version 2 (Latest Version):

https://youtu.be/RnkUMHaQTeg

We took screenshots of the game to demonstrate that the gamed passed the testing and is fully functional.

Menu Enter World

Galaxy view intro

Play

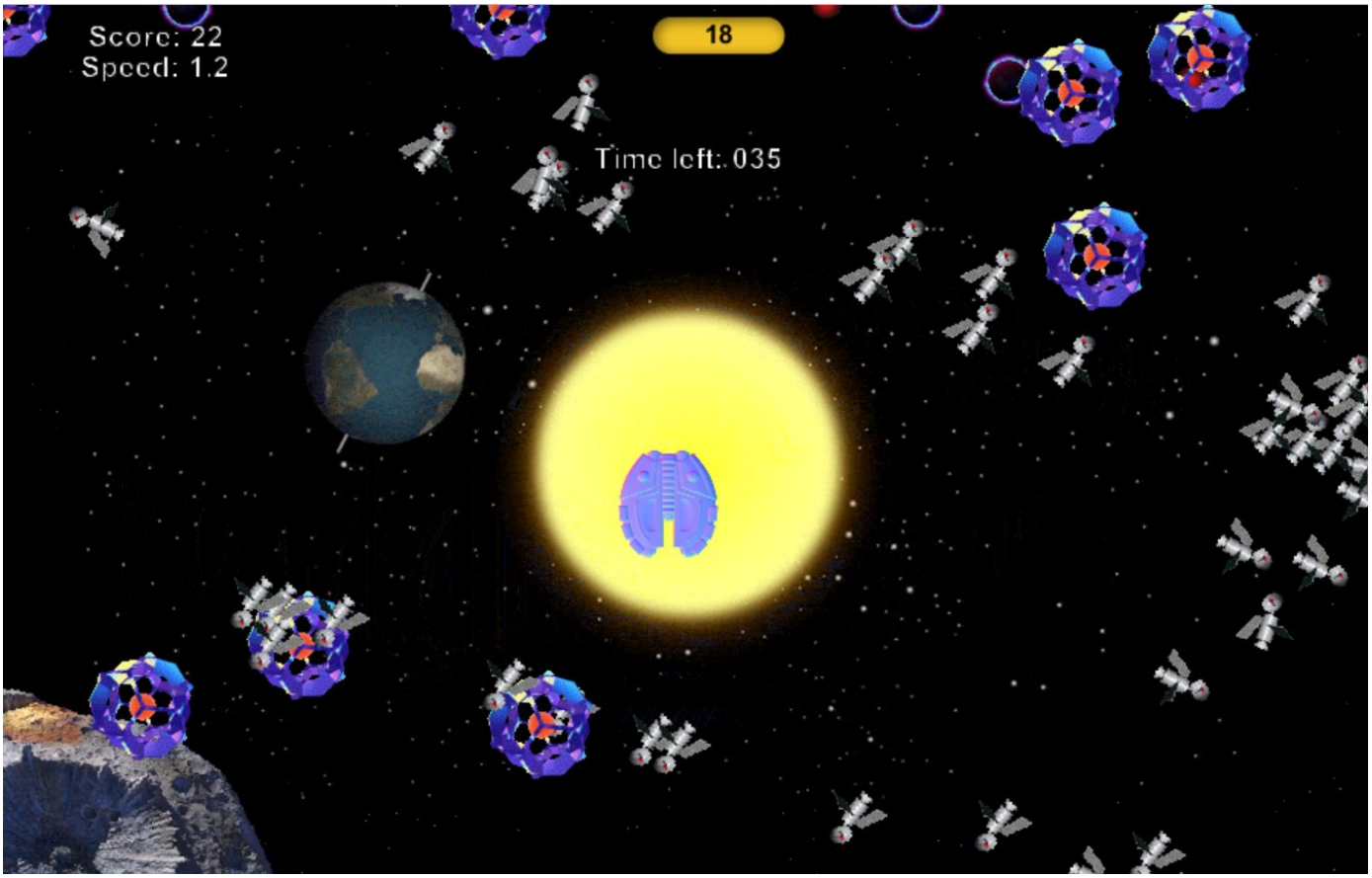
Alert

Remove Asteroid

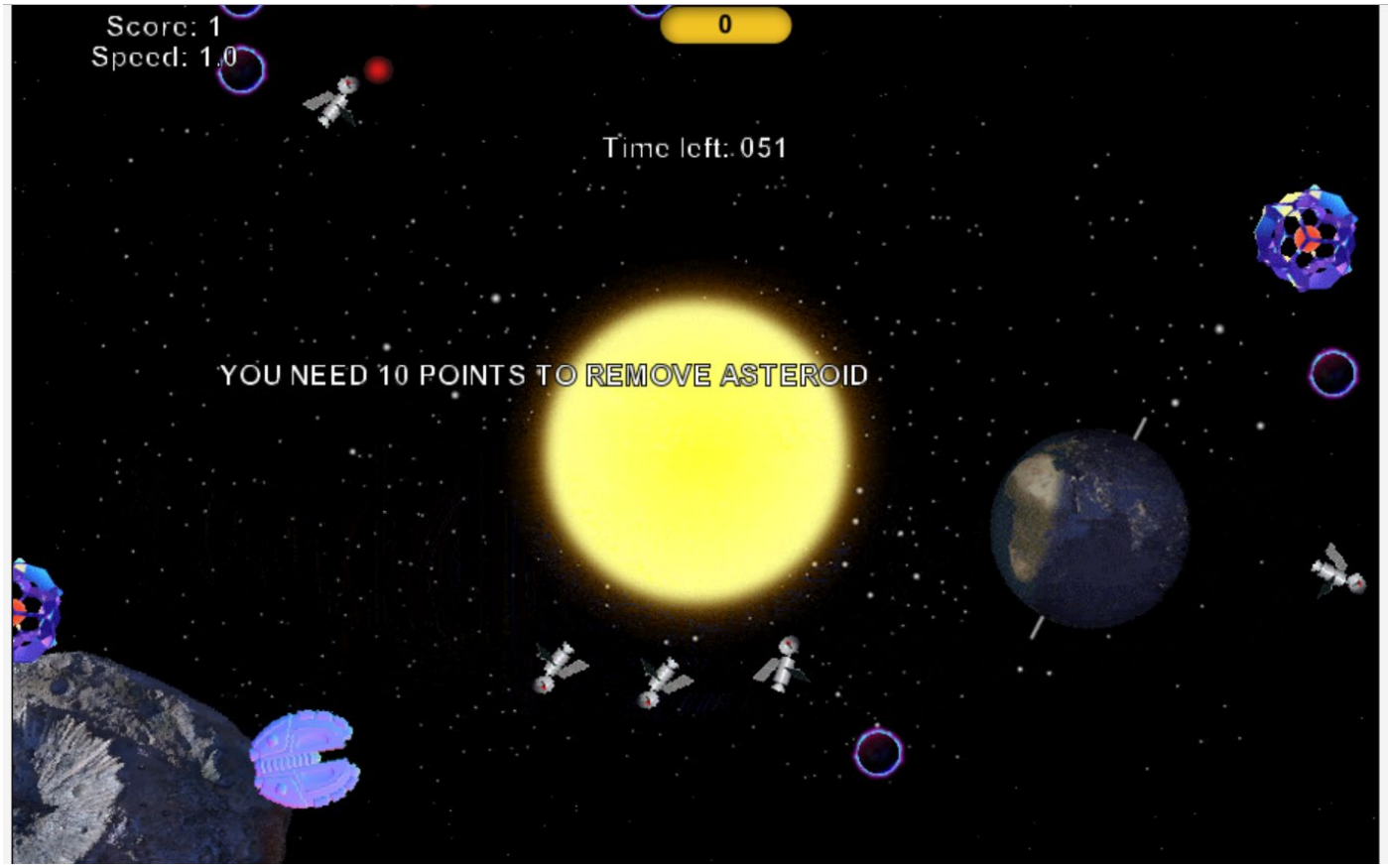

Score: 174
WELCOME TO META UNIVERSE
Time left: 22
KILL MORE INVADERS

Single Player Intro

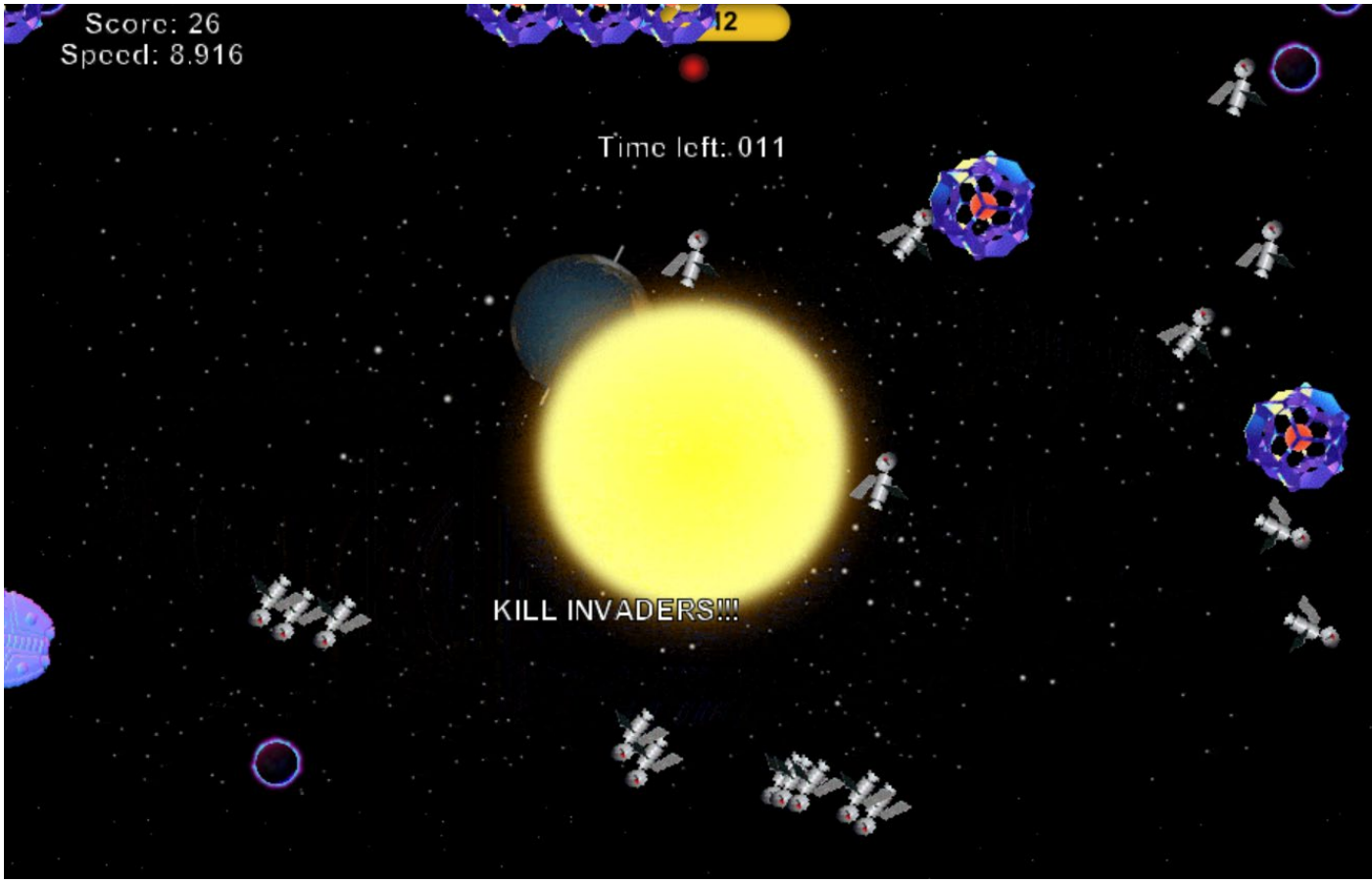
Play



Alert Instructions

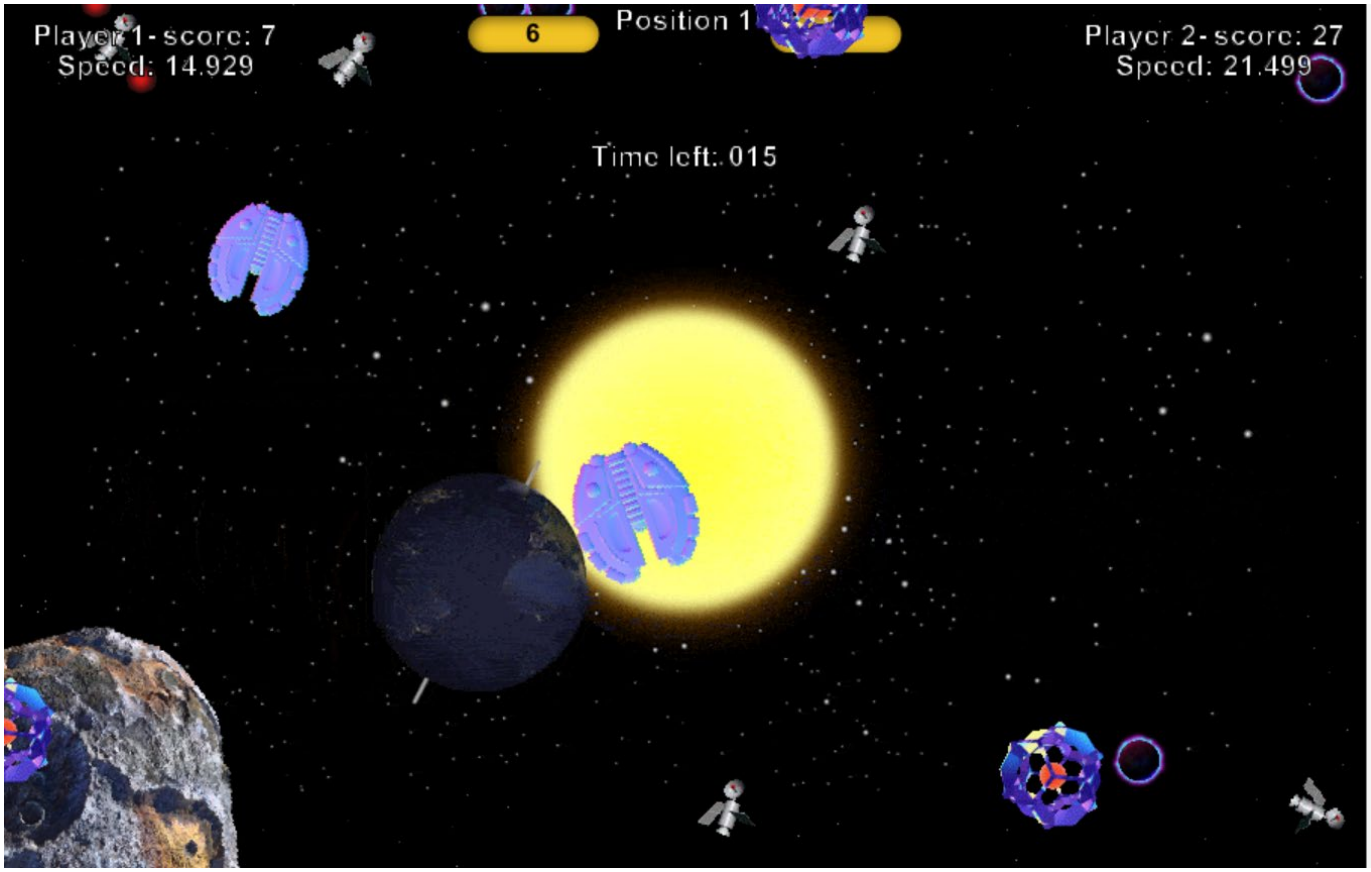


Removing Asteroid



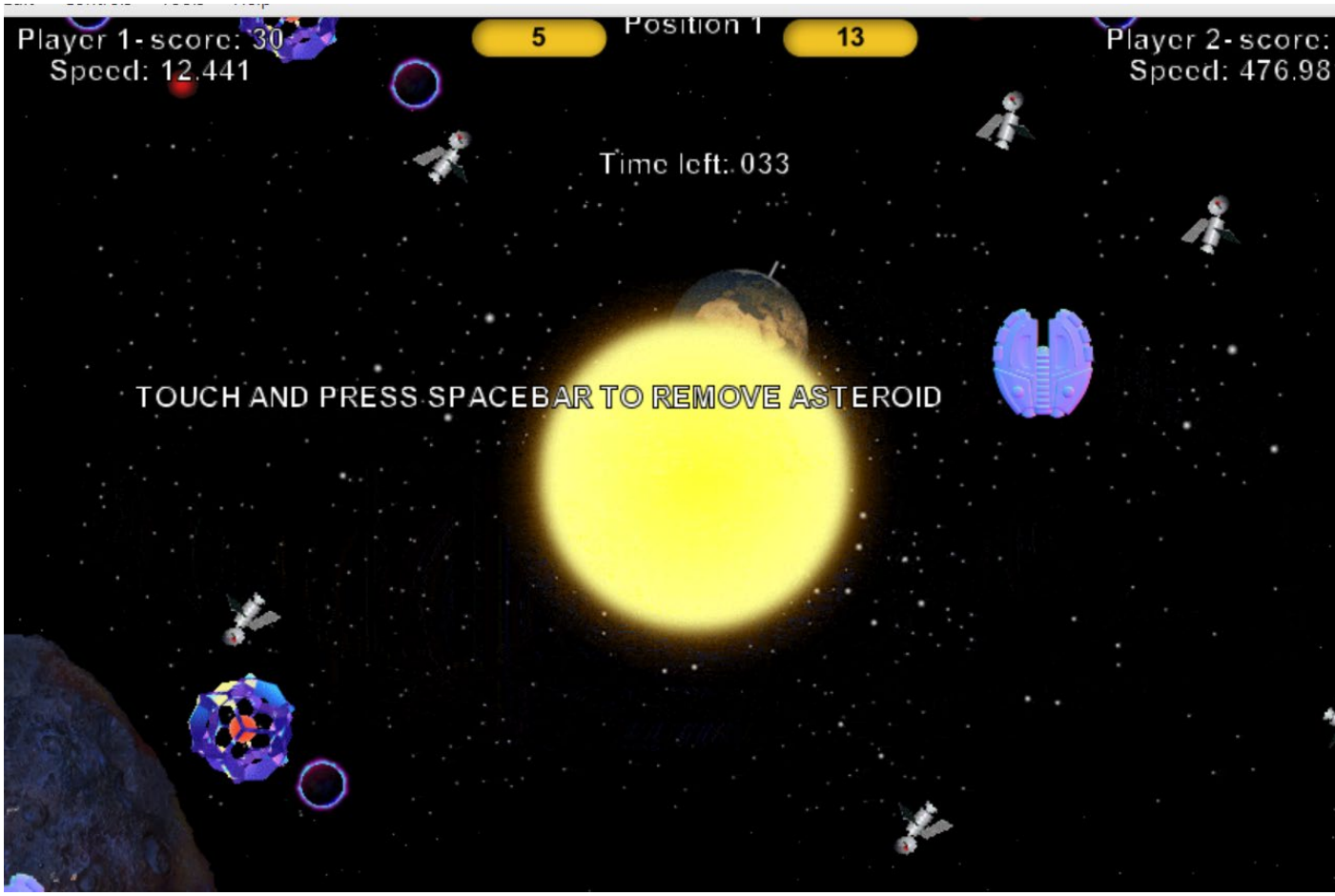
Multiplayer Intro

Play

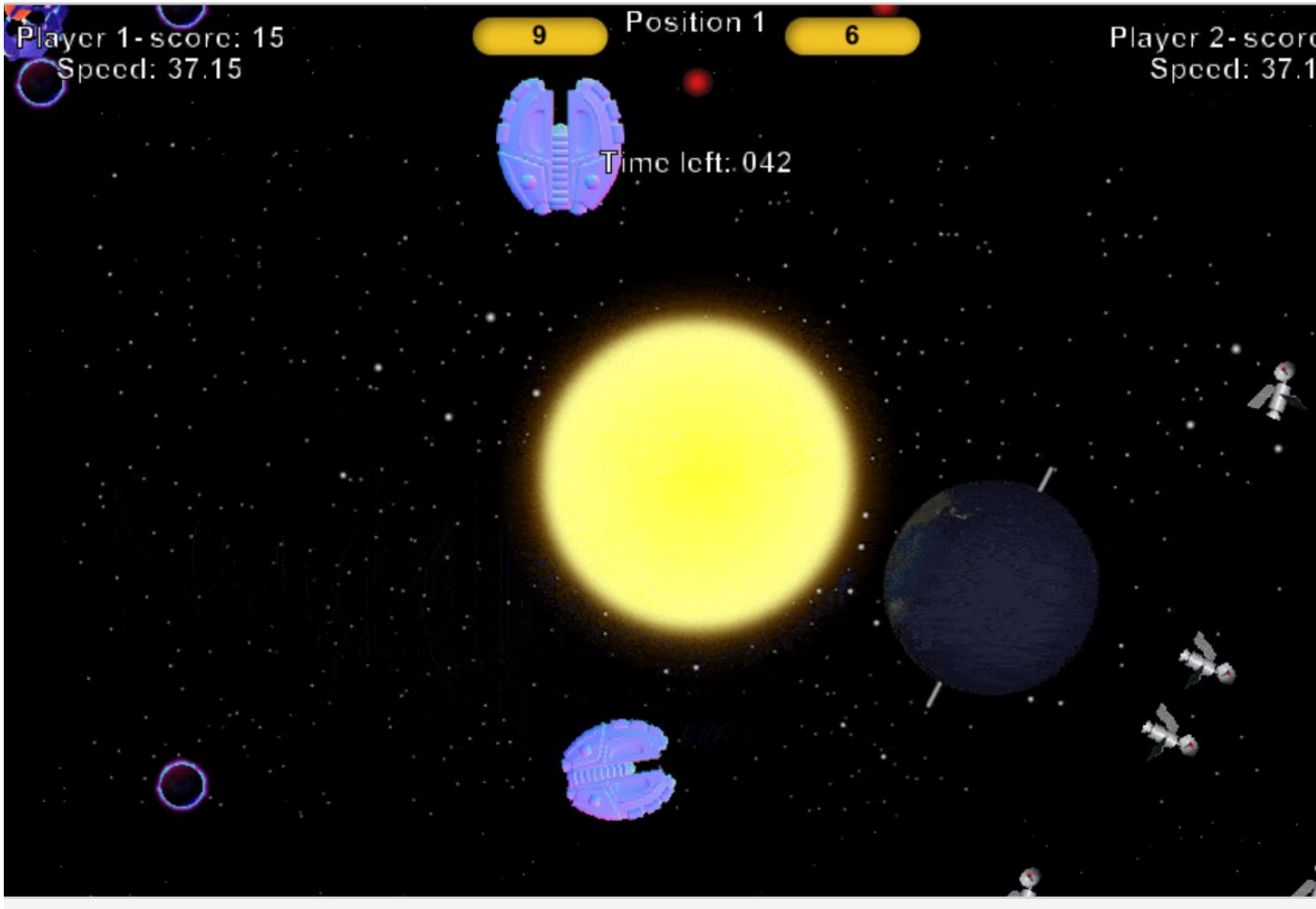


Alert Instructions

Space Bar Instructions alert


Player 1 - score: 30
Speed: 12.441
5
Position 1
13
Player 2 - score:
Speed: 476.98
Time left: 033
TOUCH AND PRESS SPACEBAR TO REMOVE ASTEROID

Removing Asteroid



EVALUATION

The **Metauniverse** Game project was a challenging task because we had to develop the game for the first time using green foot and to create a game that respect all members ideas.

It was challenging but we both enjoyed the journey to the end of it even if involved a lot of research about Green foot and learning how to use java and use pre made green foot methods inside our classes.

More likely there are thinks that we can improve but that will need us to gain more experience coding in green foot and discover different methods and effects that we can create by spending more time on testing code variables.