The Furious Farmer



Revision: 0.0.0

Final Project Report

Game Design Development

COMP6205 – Computer Graphics

Computer Science Program

Bina Nusantara University International

2016/2017

License

If you use this in any of your games. Give credit in the GDD (this document) to Alec Markarian and Benjamin Stanley. We did work so you don’t have to.

Feel free to Modify, redistribute but **not sell** this document.

TL;DR - Keep the credits section of this document intact and we are good and do not sell it.

**Table of Content**

[Overview](#_Toc483661978)

[Theme / Setting / Genre](#_Toc483661980)

[Project Description](#_Toc483661981)

[Core Gameplay Mechanics Brief](#_Toc483661982)

[Targeted platforms](#_Toc483661983)

[Project Scope](#_Toc483661984)

[Story and Gameplay](#_Toc483661985)

[Story](#_Toc483661986)

[Gameplay](#_Toc483661987)

[Assets Needed](#_Toc483661988)

[- 2D](#_Toc483661989)

[- 3D](#_Toc483661990)

[- Sound](#_Toc483661991)

[Full Coding](#_Toc483661992)

[- <Put your group CS Github url/link>](#_Toc483661993)

[Screenshot of the Game](#_Toc483661994)

# Overview

## C:\Users\Cason Kang\AppData\Local\Microsoft\Windows\INetCacheContent.Word\Title.jpg

## Theme / Setting / Genre

- Casual Point Collection game

- Apple Plantation setting

## Project Description

Our group consists of two members Cason (in charge of GUI and Models) and Dhyatmika Suriyanto (in charge of game mechanics). This is a game we created for our final project in the Computer Graphics course at Binus International.

We came up with the project idea based on paper toss as our inspiration but instead of paper toss we changed it into apple toss and called it The Furious Farmer. Our game is a web based game made with Babylon.js. The Goal of the game is to throw an apple into a basket and score the most points.

To make the game more challenging we added physics, life and wind. And the user can also adjust the power of each shot. We also added sound and a lot of models to make the game more pleasing to look at aesthetically

## Core Gameplay Mechanics Brief

- Physics: the game uses the CannonJS physics plugin to simulate the gravity and wind happening inside the game

- Power: in the game the player can control the power of each throw which controls the amount of force applied to the apple.

- Life: in the game the player can miss no more than 5 shots.

- Points: each successful throw to the basket counts as one point.

- Controls: The main controls of the game are A,W,S,D,Space and R

- a,w,s and d to control the arrow direction

- Space to control power

- R to reset the ball to the original position

## Targeted platforms

- Web: This game is made to be a web based game played on browsers.

## Project Scope

- The project took about one month to complete including the model preparation and mechanics.

- Core Team

- Cason 1901521236 + Casoncase (Gitlab account)

- In charge of GUI, Models and Sound.

- Dhyatmika Suriyanto 1901521255 + dhyatmika.s (Gitlab account)

- In charge of game mechanics.

- Libraries Used

- Babylon.js

- Hand.js

- Programs used during development

- Blender

- Sublime text

- Atom

# Story and Gameplay

## Story

The story of the game is about a farmer who is harvesting his apples. Unfortunately, some of his apples turned out to be small. The farmer becomes enraged and decides to throw the apples into the basket instead of placing them into the basket slowly. After the first few throws the wind suddenly starts to become stronger making the throws more challenging. The furious farmer slowly starts to enjoy throwing the apple and decides to count how many he can get in. The more he misses the more furious he gets if the farmer misses more than 5 throws he will lose his patience and stop throwing apples.

## Gameplay

The game starts in a field with an apple aimed at the basket. There is a board for score and wind on the left top corner and life and power at the right top corner. The player’s task is to aim and throw the apple to the basket. The player will get 5 lives and each time he throws and resets the apple there is a 60% chance that the wind will change. The range of wind is from -10 to +10 a negative value means that the wind blows to the left while a positive value means that it will blow to the right. The controls of the game are a,w,s and d for aiming the apple, space for controlling the power and r to reset the ball to the original position. If the apple reaches the goal it will automatically be placed back to the original position while if it misses the player should reset the ball at the cost of one life. When the life reaches 0 the player’s final score will be shown on screen and the game will reset.

# Assets Needed

## - 2D

- Textures

- Grass.png

- Skybox

- grass\_nx.jpg

- grass\_ny.jpg

- grass\_nz.jpg

- grass\_px.jpg

- grass\_py.jpg

- grass\_pz.jpg

## - 3D

- Environmental Art Lists

- Apple.babylon

- Basket.babylon

- manyApples.babylon

## - Sound

- Sound effects

- bgm.mp3

- yay.mp3

- end.mp3

# Full Coding

### - <https://github.com/CSBinusInternational/L4BC-Group-2>

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<title>The Furious Farmer</title>

<script type="text/javascript" src="../lib/babylon.custom.js"></script>

<script type="text/javascript" src="../lib/hand-1.3.7.js"></script>

<script type="text/javascript" src="../lib/cannon2.js"></script>

<link rel="stylesheet" type="text/css" href="../lib/style\_canvas.css" />

</head>

<peach>

<canvas id="canvas"></canvas>

<script>

var canvas = document.getElementById("canvas");

var engine = new BABYLON.Engine(canvas, true);

// var createScene = function(){

var scene = new BABYLON.Scene(engine);

var physicsPlugin = new BABYLON.CannonJSPlugin();

scene.enablePhysics(new BABYLON.Vector3(0, -9.81, 0), physicsPlugin);

var camera2;

camera2 = new BABYLON.ArcRotateCamera("camera2", BABYLON.Tools.ToRadians(270), BABYLON.Tools.ToRadians(45), 20, new BABYLON.Vector3.Zero(), scene);

camera2.setPosition(new BABYLON.Vector3(0,35,-70));

camera2.attachControl(canvas, false);

var light = new BABYLON.HemisphericLight("light1", new BABYLON.Vector3(0,5,0), scene);

light.intensity = 0.9;

var ground = BABYLON.Mesh.CreateGround("ground", 1000, 1000, 2, scene);

ground.setPhysicsState({impostor: BABYLON.PhysicsEngine.BoxImpostor, mass: 0, restitution: 0.5, friction: 0.1});

var arrow = BABYLON.Mesh.CreateBox("arrow", 0.5, scene);

arrow.scaling.z = 20;

arrow.position = new BABYLON.Vector3(0, 1, 0);

arrow.setPivotMatrix(BABYLON.Matrix.Translation(0, 0, 0.25));

var arrowhead = BABYLON.MeshBuilder.CreateCylinder("arrowhead", {diameterTop: 1.5, height: 1.5, diameterBottom: 0, tessellation: 4}, scene);

arrowhead.rotation.x = BABYLON.Tools.ToRadians(-90);

arrowhead.position.z = 5.5/20;

arrowhead.scaling.y = 1/20;

arrowhead.parent = arrow;

var basket = BABYLON.MeshBuilder.CreateCylinder("basket", {diameterTop: 27.5, height: 21, diameterBottom: 21, tessellation: 16}, scene);

basket.position = new BABYLON.Vector3(0, 10, 150)

basket.setPhysicsState({impostor: BABYLON.PhysicsEngine.CylinderImpostor, mass: 0, restitution: 0.5, friction: 10});

var goal = BABYLON.MeshBuilder.CreateCylinder("goal", {diameterTop: 26.5, height: 1, diameterBottom: 26.5, tessellation: 16}, scene);

goal.position = new BABYLON.Vector3(0, 20.7, 150)

goal.setPhysicsState({impostor: BABYLON.PhysicsEngine.CylinderImpostor, mass: 0, restitution: 0.5, friction: 10});

var wall = BABYLON.Mesh.CreateBox("wall", 5, scene);

wall.position = new BABYLON.Vector3(0, 2.5, 4.3);

var wall2 = BABYLON.Mesh.CreateBox("wall2", 5, scene);

wall2.position = new BABYLON.Vector3(-4.3, 2.5, 0);

var wall3 = BABYLON.Mesh.CreateBox("wall3", 5, scene);

wall3.position = new BABYLON.Vector3(4.3, 2.5, 0);

var wall4 = BABYLON.Mesh.CreateBox("wall4", 5, scene);

wall4.position = new BABYLON.Vector3(0, 6.5, 0);

var wall5 = BABYLON.Mesh.CreateBox("wall5", 5, scene);

wall5.position = new BABYLON.Vector3(0, 2.5, -4.3);

wall.setPhysicsState({impostor: BABYLON.PhysicsEngine.BoxImpostor, mass: 0, restitution: 0.1, friction: 10});

wall2.setPhysicsState({impostor: BABYLON.PhysicsEngine.BoxImpostor, mass: 0, restitution: 0.1, friction: 10});

wall4.setPhysicsState({impostor: BABYLON.PhysicsEngine.BoxImpostor, mass: 0, restitution: 0.1, friction: 10});

wall3.setPhysicsState({impostor: BABYLON.PhysicsEngine.BoxImpostor, mass: 0, restitution: 0.1, friction: 10});

wall5.setPhysicsState({impostor: BABYLON.PhysicsEngine.BoxImpostor, mass: 0, restitution: 0.1, friction: 10});

var materialInvisible = new BABYLON.StandardMaterial("material1",scene);

materialInvisible.alpha = 0;

basket.material = materialInvisible;

goal.material = materialInvisible;

wall2.material = materialInvisible;

wall3.material = materialInvisible;

wall4.material = materialInvisible;

wall.material = materialInvisible;

wall5.material = materialInvisible;

var material2 = new BABYLON.StandardMaterial("material2",scene);

material2.diffuseColor = BABYLON.Color3.Red();

material2.emmisiveColor = BABYLON.Color3.Red();

material2.alpha = 1.0;

var materialBlue = new BABYLON.StandardMaterial("material2",scene);

materialBlue.diffuseColor = BABYLON.Color3.Blue();

materialBlue.emmisiveColor = BABYLON.Color3.Blue();

materialBlue.alpha = 1.0;

var materialBasket = new BABYLON.StandardMaterial("materialBasket", scene);

materialBasket.diffuseColor = new BABYLON.Color3(42, 21, 21);

materialBasket.emmisiveColor = new BABYLON.Color3(165, 42, 42);

materialBasket.specularPower = 100000;

materialBasket.alpha = 1.0;

var groundMaterial = new BABYLON.StandardMaterial("groundMaterial", scene);

groundMaterial.diffuseTexture = new BABYLON.Texture("texture/grass.png", scene);

groundMaterial.specularPower = 10000;

groundMaterial.diffuseTexture.uScale = 5.0;

groundMaterial.diffuseTexture.vScale = 5.0;

ground.material = groundMaterial;

//importing apple

var peach;

var peachImpostor;

BABYLON.SceneLoader.ImportMesh("","assets/", "apple.babylon", scene, function (newMeshes, particleSystems) {

peach = newMeshes[0];

console.log(peach.scaling.x);

peach.scaling = new BABYLON.Vector3(40, 40, 40);

peach.position = new BABYLON.Vector3(0, 3, 0);

peachImpostor = peach.setPhysicsState({impostor: BABYLON.PhysicsEngine.SphereImpostor, mass: 5, restitution: 0.1, friction: 1, linearDamping: 0.5});

peach.material = material2;

});

//importing basket

var basketModel;

BABYLON.SceneLoader.ImportMesh("","assets/", "basket.babylon", scene, function (newMeshes, particleSystems) {

var thing = newMeshes[2];

thing.position = new BABYLON.Vector3(0, -10, 0);

basketModel = newMeshes[1];

basketModel.scaling = new BABYLON.Vector3(0.1, 0.1, 0.12);

basketModel.position = new BABYLON.Vector3(0, 0, 150);

basketModel.material = materialBlue;

});

//importing apples inside basket

var manyApples;

BABYLON.SceneLoader.ImportMesh("","assets/", "manyApples.babylon", scene, function (manyApples, particleSystems) {

manyApples = manyApples[0];

manyApples.scaling = new BABYLON.Vector3(75, 75, 75);

manyApples.position = new BABYLON.Vector3(-6.3, 20, 149);

manyApples.rotation.z = BABYLON.Tools.ToRadians(-35);

manyApples.rotation.x = BABYLON.Tools.ToRadians(-99);

manyApples.material = material2;

});

//skybox

var skybox = BABYLON.Mesh.CreateBox("skybox", 4000.0, scene);

var skyboxMaterial = new BABYLON.StandardMaterial("skybox", scene);

skyboxMaterial.backFaceCulling = false;

skyboxMaterial.reflectionTexture = new BABYLON.CubeTexture("assets/skybox/grass", scene);

skybox.infiniteDistance = true;

skyboxMaterial.diffuseColor = new BABYLON.Color3(0, 0, 0);

skyboxMaterial.specularColor = new BABYLON.Color3(0, 0, 0);

skyboxMaterial.reflectionTexture.coordinatesMode = BABYLON.Texture.SKYBOX\_MODE;

skybox.material=skyboxMaterial;

scene.registerBeforeRender(function () {

if (peach) {

if(peach.intersectsMesh(goal, false)){

//increase point

var yay = new BABYLON.Sound("yay", "assets/sound/yay.mp3", scene, null, { loop: false, autoplay: true, volume: 0.5});

point++;

score.text = "Score: "+point.toString()+ " || Wind: "+wind.toString();

//reset peach

wall.position.y = 2.5;

wall2.position.y = 2.5;

wall3.position.y = 2.5;

wall4.position.y = 6.5;

wall5.position.y = 2.5;

wall.updatePhysicsBodyPosition();

wall2.updatePhysicsBodyPosition();

wall3.updatePhysicsBodyPosition();

wall4.updatePhysicsBodyPosition();

wall5.updatePhysicsBodyPosition();

peach.position = new BABYLON.Vector3(0, 2.5, 0);

peach.updatePhysicsBodyPosition();

//return arrow

arrow.position.y = 1;

//reset wind

phyEng.setGravity(new BABYLON.Vector3(0, -9.81, 0));

//reset speed

speed = 50;

}

}

});

// return scene;

// }

// var scene = createScene();

var switcher = 1;

var map = {};

var point = 0;

var speed = 50;

//adjust direction of throw

var angle = 0;

var anglevertical = 45;

var phyEng = scene.getPhysicsEngine();

var thrown = false;

var wind = 0;

var change = 0;

var incr = 5;

var life = 5;

//ScoreBoard

var w = window.innerWidth;

var h = window.innerHeight;

var scoreBoard = new BABYLON.ScreenSpaceCanvas2D(scene, {

id: "canvas2", parent: canvas,x:0, y:h-100,

size: new BABYLON.Size(300, 100),

backgroundFill: "#4040408F",

backgroundRoundRadius: 50,

});

var score = new BABYLON.Text2D("Score: "+point.toString()+ " || Wind: "+wind.toString(), {

parent: scoreBoard,

id: "score",

marginAlignment: "h: center, v:center",

fontName: "20pt Arial",

});

//Power Board

var powerBoard = new BABYLON.ScreenSpaceCanvas2D(scene, {

id: "canvas3", parent: canvas,x:w-300, y:h-200,

size: new BABYLON.Size(300, 100),

backgroundFill: "#4040408F",

backgroundRoundRadius: 50,

});

var power = new BABYLON.Text2D("Power: 0", {

parent: powerBoard,

id: "speed",

marginAlignment: "h: center, v:center",

fontName: "20pt Arial",

});

//Life Board

var lifeBoard = new BABYLON.ScreenSpaceCanvas2D(scene, {

id: "canvas4", parent: canvas,x:w-300, y:h-100,

size: new BABYLON.Size(300, 100),

backgroundFill: "#4040408F",

backgroundRoundRadius: 50,

});

var nyawa = new BABYLON.Text2D("Life: 5", {

parent: lifeBoard,

id: "nyawa",

marginAlignment: "h: center, v:center",

fontName: "20pt Arial",

});

//Background music

var bgm = new BABYLON.Sound("BGM", "assets/sound/bgm.mp3", scene, null, { loop: true, autoplay: true, volume: 0.05});

//Game Input

window.onkeydown = window.onkeyup = function(event){

event = event || window.event;

map[event.keyCode] = event.type == 'keydown';

//W

if(map[87] && anglevertical < 85){

anglevertical+=2

}

//S

if(map[83] && anglevertical > 0){

anglevertical-=2

}

//A

if(map[65] && angle > -90){

angle-=1

}

//D

if(map[68] && angle < 90){

angle+=1

}

//space - throwing

if(map[32]){

//increase power while held

if(speed >= 350){

incr = incr \* -1;

}

else if(speed <= 0){

incr = incr \* -1;

}

speed += incr;

//Changing the speed into percent

var percent = Math.floor((speed/350)\*100);

power.text="Power: "+percent.toString()+"%";

wall.position.y = -100;

wall2.position.y = -100;

wall3.position.y = -100;

wall4.position.y = -100;

wall5.position.y = -100;

}else if(map[32] == false){

//when released, launch ball

if(peach.position.z <= 1 && speed > 50){

//remove arrow

arrow.position.y = -100;

//apply wind

phyEng.setGravity(new BABYLON.Vector3(wind, -9.81, 0));

thrown = true;

//finding throw direction

var radangle = BABYLON.Tools.ToRadians(angle);

var radanglevertical = BABYLON.Tools.ToRadians(anglevertical);

var direction = new BABYLON.Vector3(speed \* Math.sin(radangle), speed \* Math.tan(radanglevertical), speed \* Math.cos(radangle));

direction.scaleInPlace(speed / Math.sqrt(speed \* Math.sin(radangle) \* speed \* Math.sin(radangle) + speed \* Math.tan(radanglevertical) \* speed \* Math.tan(radanglevertical) + speed \* Math.cos(radangle) \* speed \* Math.cos(radangle)));

//throwing

peach.applyImpulse(direction, peach.position);

//reset speed

speed = 50;

}

}

//R - reset object

if(map[82]){

//remove then create new peach object

score.text = "Score: "+point.toString()+ " || Wind: "+wind.toString();

wall.position.y = 2.5;

wall2.position.y = 2.5;

wall3.position.y = 2.5;

wall4.position.y = 6.5;

wall5.position.y = 2.5;

wall.updatePhysicsBodyPosition();

wall2.updatePhysicsBodyPosition();

wall3.updatePhysicsBodyPosition();

wall4.updatePhysicsBodyPosition();

wall5.updatePhysicsBodyPosition();

peach.position = new BABYLON.Vector3(0, 2.5, 0);

peach.updatePhysicsBodyPosition();

//return arrow

arrow.position.y = 1;

//remove wind

phyEng.setGravity(new BABYLON.Vector3(0, -9.81, 0));

//reset speed

speed = 50;

function reloadPage()

{

location.reload();

}

//reduce life

life--;

//game over screen

if(life==0){

bgm.stop();

var end = new BABYLON.Sound("gameEnd", "assets/sound/end.mp3", scene, null, { loop: false, autoplay: true, volume: 0.5});

var gameOver = new BABYLON.ScreenSpaceCanvas2D(scene, {

id: "canvas5", parent: canvas,x:0, y:0,

size: new BABYLON.Size(w, h),

backgroundFill: "#404040FF",

});

var finscore = new BABYLON.Text2D("Your Final Score: "+point.toString(), {

parent: gameOver,

id: "finscore",

marginAlignment: "h: center, v:center",

fontName: "20pt Arial",

});

setTimeout(function(){

reloadPage();}, 4000);

}

nyawa.text="Life: "+life.toString();

}

}

engine.runRenderLoop(function(){

if(thrown){

thrown = false;

change = Math.random()\*10;

if(change <= 6){

wind = Math.floor(Math.random()\*20.9)-10;

}

}

arrow.rotation.y = BABYLON.Tools.ToRadians(angle);

arrow.rotation.x = BABYLON.Tools.ToRadians(-anglevertical);

scene.render();

});

window.addEventListener("resize", function(){

engine.resize();

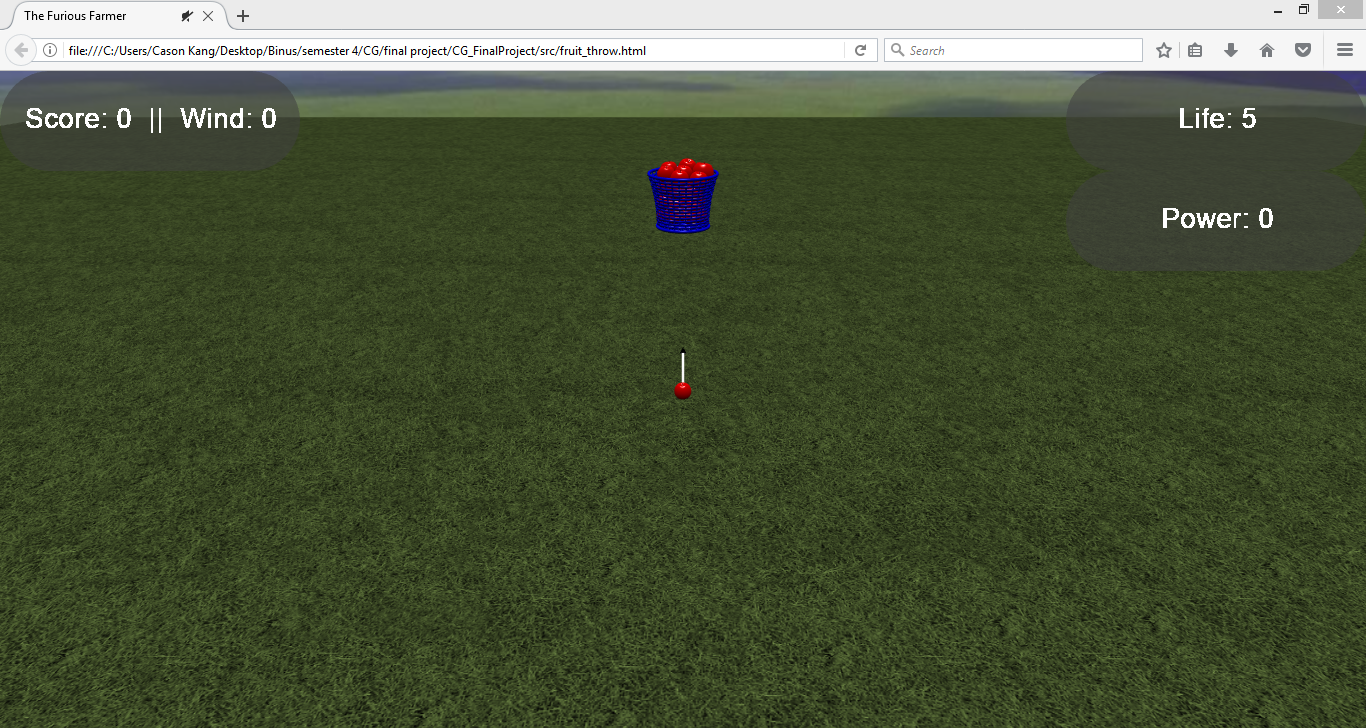
});

</script>

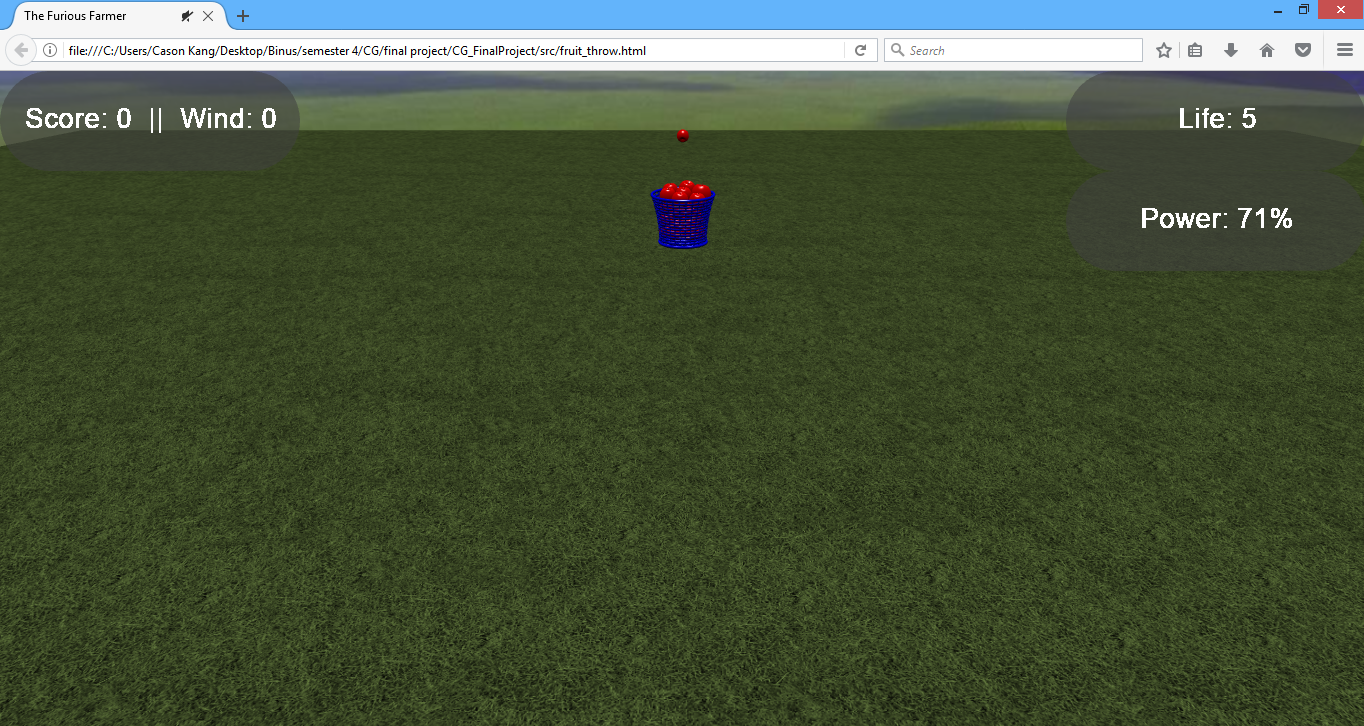
</peach>

</html>

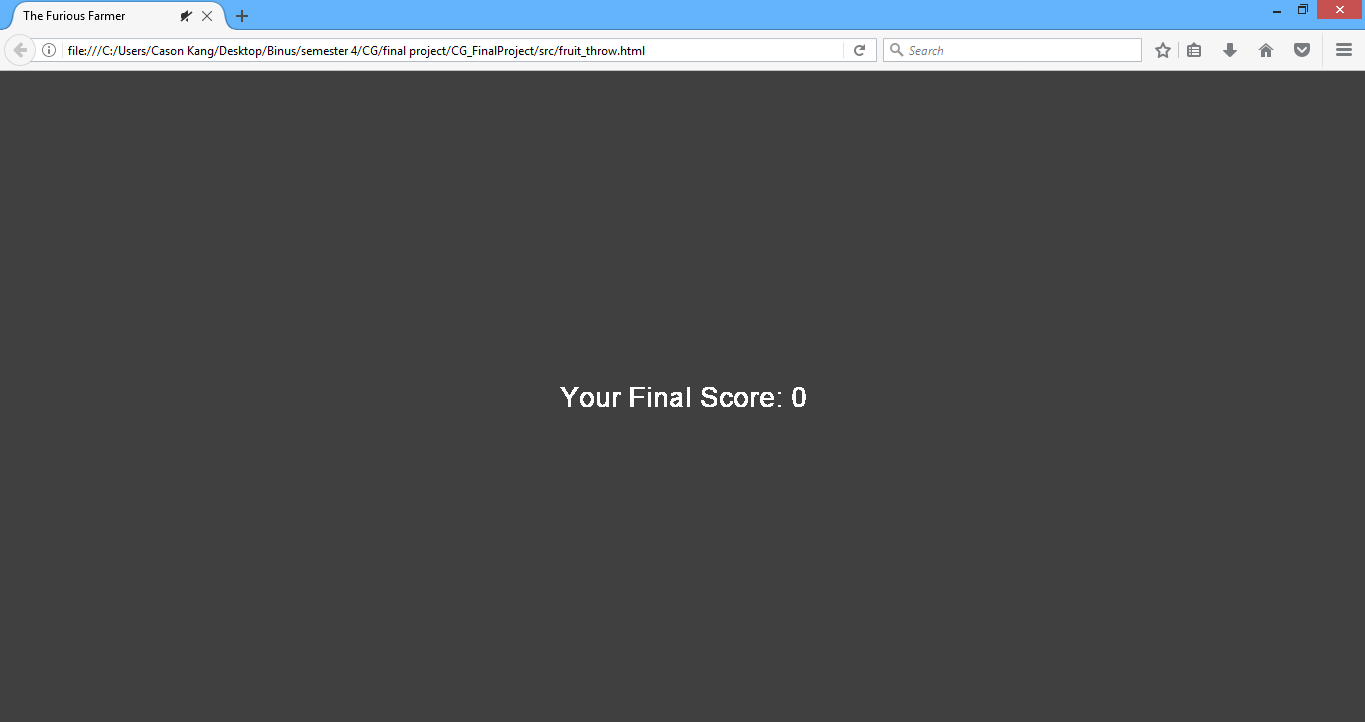
# Screenshot of the Game



Start Screen



Gameplay



End Screen