

**ISE491 & COME 491  
 Project**

**Games are the keys to make us happier inside same as love**

**Advisor:Zeynep Hilal KILIMCI**

**Prepared By: Tamer İnce**

**201435007**

**&**

**Anil Karlidağ**

**201431014**

**PROJECT COMPONENTS**

**1-)Project purpose and targets**

Understanding Gamer Psychology,Why Do People Play Games?

We all have basic psychological needs .For example when your mind is fully occupied by other things and you need a break

Games have been played for a long, long time now. Remember those countless hours we spent playing Pacman on the arcades or playing Pokemon as children? For sure we do .

concerning the possible negative health effects of gaming, it is not often highlighted the positive effects of video gaming, what is so appealing about games, and why we play and continue to play. what motivates us to play games.

**Invisible Needs**

**Escapism (mostly seemed a person need a break time not to think about responsibilities )**

With video games, you can escape the perpetual boredom of the real world, and become anything you want to be

Gamers often throw around the term “**escapism**” when talking about their hobby, but this is a hollow explanation for what actually motivates us to play games. In fact, the word “escape” contains some negative implications – suggesting that those who play games feel a need to break free from the mundane slavery of their reality. We enjoy retreats to other realities – ones more fantastical than our own – but we aren’t always driven to play games because we are trying to escape our lives. The real motivations for play are far more complex, and games fulfill several real-world human needs in a number of positive ways.

We all have basic psychological needs .For example when your mind is fully occupied by other things and you need a break when **we’re at work**, or when **we’re engaging** in a, or on a **final exams week** for the students.while we are -playing a video game. These needs are always operating.

**Competence & Emotional satisfaction**  **(Feeding the secret Ego inside )**

Some People like to prove themself by achiving somethings and being better then others or seeing themselves getting better on doing things.

Complex needs-satisfaction metrics narrow down to three basic categories. The first of these needs is a need for **competence** – that is a desire to seek out control or to feel mastery over a situation. People like to feel successful, and we like to feel like we’re growing and progressing in our knowledge and accomplishments. This need plays out in real life when people decide to switch careers or go back to school because their current job isn’t rewarding or challenging enough. It’s also easy to see how video games make us feel more accomplished. Every time we level up in Any Game or defeat a challenging boss in the game, games are fulfilling our desire to feel competent.

**Boredom**

People live in the “Work eat and sleep” Cycle but to break that cycle of boredom as I call people need to have fun sometimes

Games are people’s fall back way to kill time of their free time by having fun and also a kind of way to clear mind from RL ( Real life) when they have nothing to do.Instead of reading boks or other hobbies playing is a great way to let the time pass.Gaming is also a kind of hobby

**Difficulty making friends IRL(İn Real Life)**

Some people may not be as much social as others and in this case games are eligable thing to gather a community together.Some games can be played and get progresssed by having an alliance or groupmates.”You cant do it alone” The life is also similar you should not live and do things by yourself as the not being alone key of enjoying the life also a key for otherthings.Beating the bosses with your best friends also counted.

**Relaxing & Stress relief**

When we are stressed with homeworks exams or projects games are perfect way to get our mind away from these things .

Games perfectly target most of these needs to make us happier inside

**2-) Time plan**

**TASK Start Date Duration End Date**

1)Sellecting A project 1. Week of September 1week 2.week of September

2)Advisor Meetings 18th October 7weeks 6th December

3)DoingProject meetings 18th October 10weeks 30th December

4)Progress Presentation 29th November 1Hour 29th Novembe

**Gann Chart**

1 =========🡺

2 ============🡺

3 ===========================🡺

4 =🡺

\_\_\_\_|\_\_\_\_\_\_\_\_\_\_|\_\_\_\_\_\_\_\_\_\_\_\_\_\_|\_\_\_\_\_\_\_\_\_\_\_\_\_\_|\_\_\_\_\_\_\_\_\_\_\_\_\_|\_\_\_\_\_\_\_\_\_\_\_\_\_\_|\_\_\_\_\_\_\_\_4th Sep 11th Sep 29th Nov 18th Oct 6th Dec 30th Dec

**3-)Advisor Meeting Table**

First Meeting 18/10/2018 1 hour

Second Meeting 01/11/2018 1 hour

Third Meeting 19/11/2018 2 hours

Fourth Meeting 13/12/2018 1 hour

Summary Of the First Meeting:

We discussed out project about its contents.What should our game has or has not?To Sum up.We Decided to make the game endless style,Game will be played horizontally.We going to use UNİTY as the patform and C# language.The other contents we should add to the game are Music(optional on/off) BackGround. İnstead of earth optinal choices.

Summary Of the Second Meeting:

We decided to ad more components to our game after a discussion what else a game should has?.To Pause the game button,Levels,Optinal Space Ships,Score,log

Summary Of the Third Meeting:

Basicly , as a team we have mentioned about what we are aiming to do for the term project and what will it take to do it and how we will .We are going to an Android Game that works on mobile devices only .We will use UNİTY game engine and C# for the language .Always C# language is being suggested for UNİTY platform. We are planning to do leveled space shooting game .On the first level its free for you to chose appearance but after that you need to use your scores that you collected on the first level to open the other levels and appearances . After passing to another level if you fail we might lose the score you got and you will go back to previous level .By making the user dropping off level we believe that will make the user connect to game more and play more and get more scores. User can control not only horizontally and vertically , user will be able to control the spaceship’s looking angle for 360 degree. It will help user to handle with the difficulties with an enjoyable way and enjoy the game more

Summary Of the Fourth Meeting:

After 4-5 weeks of working on the project , We showed our Mobile Game Codes and and The Game itself to our advisor.Our advisor and We deeply examine the codes and game playing how else it could be, is everything ok?,is there any mistakes on the code , what thing we might do to make the game code more efficient for players to play smooth.We showed to our advisor the game levels and explained how it Works

**4-)Requirements Analysis Document1. Introduction**

We are planning to do a Mobile Game Project(MGP) by using UNITY as platform and as the language C#.Our Game is for AndoidOS.All Devices those have Android Operatin System can open this game.

**1.1 Purpose of the system**

The Purpose of the System is to let the player enjoy playing and to feel how is the feeling when you face with many difficulties and succeed to overcome.Another purpose is to increase players’ reflexes to handle wıth the things unexpectedly happen

**1.2 Scope of the system**

This game’s field of activity is in Entertaintment.We aim to make a game to fill our clients’ empty time with joyness.

**1.3 Objectives and success criteria of the Project**

Succes criterias are to see many people downloaded the game and to know that people are enjoying to play our game

**1.4 Definitions, acronyms, and abbreviations**

MGP(Mobile Game Project)

**1.5 References**

The Lessons we took apart in the Dogus University,Internet,UNITY Document, https://docs.unity3d.com/Manual/index.html

**1.6 Overview**

A Mobile Game thats made by using UNITY and C#

**2. Current system**

Player has controller for the movements

**3. Proposed system**

Try to avoid difficulties and stay alive

**3.1 Overview**

In our Mobile gamet he player will try to stay alive by facing with mant difficulties

**3.2 Functional requirements**

Playing the game smoothly , Opening the game immedately Not LATELY,Game should not freze while playing,The Game’s Size should not be very big

**3.3 Nonfunctional requirements**

**3.3.1 Usability**

Playing games will help people not to think about their Daily problems,Filling free times with joyness,To entertaint the people especially Game Addicted ones

**3.3.2 Reliability**

The improvements in the game should not dissapper when you fail

**3.3.3 Performance**

While playing the game , game should not be frozen and can be played smothly

**3.3.4 Supportability**

If there would be purchasing in the game , developers should ensure about the game mistakes are not exist on buying and to make the buyings by players surely

**3.3.5 Implementation**

The new levels can be added in to the game and Game can be updated to let the player play faster and smootly

**3.3.6 Interface**

Game Menu

**3.4 System models**

**3.4.1 Scenarios**

Surviving in Space

**3.4.2 Use case model**

Open game and click play button

**3.4.3 Object model**

Object Orianted Modelling

**3.4.5 User interface—navigational paths and screen mock-ups**

Game Menu to chose player objects

**4. Glossary**

**5-)Documentation and User Manual**

**DusmanMermiKodu.cs**

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class düsmanMermiKodu : MonoBehaviour {

Rigidbody fizik;

void Start () {

fizik = GetComponent<Rigidbody>();

fizik.velocity = new Vector3(-15f, 0f, 0f);

}

}

**DüsmanGemiKodu.cs**

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

using UnityEngine.UI;

using UnityEngine.SceneManagement;

public class düşmanGemiKodu : MonoBehaviour {

Rigidbody fizik;

public GameObject mermi;

public Transform mermiPos;

float zaman = 0f;

GameObject gemiPlayer1;

int darbeSayısı = 0;

//public GameObject patlamaEfekti;

GameObject oyunKontroller;

public ParticleSystem patlama;

// public ParticleSystem sonPatlama;

AudioSource[] sesler;

public Image canBarı;

float oyunbas = 0f;

//public GameObject bigEXP;

public GameObject mysprite;

Collider mycoll;

bool ptlamaEfktAçık = false;

//public GameObject didIt;

bool mermiAtsın = true;

void Awake()

{

oyunKontroller = GameObject.FindGameObjectWithTag("GameController");

}

void Start () {

fizik = GetComponent<Rigidbody>();

InvokeRepeating("mermiAt", 1f, 3f);

canBarı.fillAmount = 1f;

// gemiPlayer1 = GameObject.FindGameObjectWithTag("gemi1Sprite");

Invoke("gemiPlayerBulucu", 0.1f);

sesler = GetComponents<AudioSource>();

mycoll = GetComponent<Collider>();

}

void Update()

{

zaman = zaman + Time.deltaTime;

if (zaman > 0.5f)

{

enemiyoluyap();

}

//if (gemiPlayer1!=null) //sorun burada gemiplayer1 i bulamıor instantied edildikten sonra fix it

//{

// enemiyoluyap();

//}

// enemiyoluyap();

}

void enemiyoluyap()

{

if (transform.position.y < gemiPlayer1.transform.position.y)

{

fizik.velocity = new Vector3(0f, 0.5f, 0f);

if (gemiPlayer1.transform.position.y - transform.position.y < 0.1f)

{

fizik.velocity = new Vector3(0f, 0f, 0f);

}

}

if (transform.position.y > gemiPlayer1.transform.position.y)

{

fizik.velocity = new Vector3(0f, -0.5f, 0f);

Debug.Log("inior tamer");

if (transform.position.y - gemiPlayer1.transform.position.y < 0.1f)

{

fizik.velocity = new Vector3(0f, 0f, 0f);

}

}

}

void mermiAt()

{

if(mermiAtsın==true)

{

//Instantiate(mermi, mermiPos.position, Quaternion.Euler(0f,-90f,0f));

Instantiate(mermi, mermiPos.position, Quaternion.identity);

}

}

void OnTriggerEnter(Collider coll)

{

if(coll.tag=="mermitagi")

{

darbeSayısı++;

canBarı.fillAmount = canBarı.fillAmount - 0.05f;

//if(ptlamaEfktAçık==false)

// {

//

// ptlamaEfktAçık = true;

// }

//GameObject efekt = Instantiate(patlamaEfekti, transform.position, Quaternion.identity);

//Destroy(efekt, 2f);

patlama.Play();

Destroy(coll.gameObject);

if (canBarı.fillAmount<=0f)

{

//Destroy(gameObject);

// Instantiate(bigEXP, coll.transform.position, Quaternion.identity);

// sonPatlama.Play();

patlama.transform.localScale = new Vector3(5f, 5f, 5f);

patlama.Play();

sesler[0].Play();

//gameObject.SetActive(false);

mysprite.SetActive(false);

mycoll.enabled = false;

//transform.position = new Vector3(500, 500, 500);

// SceneManager.LoadScene(0);

//didIt.SetActive(true);

//sesler[1].Play();

mermiAtsın = false;

oyunKontroller.GetComponent<oyunahakimimkodu>().didİtÇal();

GameObject.FindGameObjectWithTag("gemi1Sprite").GetComponent<GemiPlayerKodu>().yıldızlarıEkle(); //bura ii durmadı burayı düzelt

// Invoke("AnaMenüReturn", 2.5f); //çünkü sadece tek bir gemi tagi için yapabilir

}

}

}

void gemiPlayerBulucu()

{

gemiPlayer1 = GameObject.FindGameObjectWithTag("gemi1Sprite");

}

public void AnaMenüReturn()

{

SceneManager.LoadScene(0);

}

}

,

**mermiKodu.cs**

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

using UnityEngine.UI;

public class mermiKodu : MonoBehaviour {

// public GameObject patlatmaEfekti;

public ParticleSystem patlamaEfekti;

AudioSource[] sesler;

// public GameObject müzikobjesi;

//float score = 0f;

GameObject gemiplayer;

GemiPlayerKodu gemiplayerKodu;

void Start () {

sesler = GetComponents<AudioSource>();

gemiplayer = GameObject.FindGameObjectWithTag("gemi1Sprite");

gemiplayerKodu = gemiplayer.GetComponent<GemiPlayerKodu>();

}

void OnTriggerEnter(Collider coll)

{

if(coll.tag=="kayatagi")

{

sesler[0].Play();

Destroy(coll.gameObject);

transform.position = new Vector3(500f, 500f, 500f);

Destroy(gameObject,1f);

//Instantiate(müzikobjesi, new Vector3(500f, 500f, 500f), Quaternion.identity);

//GameObject efktobje = Instantiate(patlatmaEfekti, coll.transform.position, Quaternion.identity);

//Destroy(efktobje, 2f);

patlamaEfekti.Play();

patlamaEfekti.transform.position = new Vector3(coll.transform.position.x, coll.transform.position.y, coll.transform.position.z);

//gemiplayer.GetComponent<GemiPlayerKodu>().score += 100; //ASLINDA BU VE ALTTAKİ SATIR ÇALIŞIOR AMA AŞADAKİNİ DENEDK HEM GET YAPMAMAK İÇİN

//gemiplayer.GetComponent<GemiPlayerKodu>().scoreumText.text = "Score" + gemiplayer.GetComponent<GemiPlayerKodu>().score;

gemiplayerKodu.score += 100;

gemiplayerKodu.scoreumText.text = "Score" +gemiplayerKodu.score;

// GameObject.FindGameObjectWithTag("gemiplayer").GetComponent<GemiPlayerKodu>().scoreText.text= "Scoreeee " + GameObject.FindGameObjectWithTag("gemiplayer").GetComponent<GemiPlayerKodu>().score;

}

}

}

**mkkübü.cs**

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class mkkübü : MonoBehaviour {

Rigidbody rb;

void Start () {

rb = GetComponent<Rigidbody>();

}

// Update is called once per frame

void Update () {

if(Input.GetMouseButtonDown(0))

{

rb.velocity = new Vector3(0f, 0f, 0f);

rb.AddForce(0, 300, 0);

}

}

}

**müzikÇalKod.cs**

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class müzikÇalKod : MonoBehaviour {

// Use this for initialization

void Start () {

Invoke("kendiniYokET", 2f);

}

// Update is called once per frame

void Update () {

}

void kendiniYokET()

{

Destroy(gameObject);

}

}

**showfps.cs**

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

using UnityEngine.UI;

public class showfps : MonoBehaviour {

float deltaTime;

public Text taşçıkmasüresitext;

void Start () {

}

void Update () {

deltaTime += (Time.deltaTime - deltaTime) \* 0.1f;

float fps = 1.0f / deltaTime;

taşçıkmasüresitext.text = Mathf.Ceil(fps).ToString();

}

}

**dontDestroy.cs**

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class dontDestroy : MonoBehaviour {

void Awake()

{

GameObject[] obj = GameObject.FindGameObjectsWithTag("music");

if(obj.Length>1) // || Application.loadedLevel>2

{

Destroy(this.gameObject);

}

else

{

DontDestroyOnLoad(this.gameObject);

}

}

void Start () {

}

}

**boundaries.cs**

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class boundaries : MonoBehaviour

{

private Vector2 screenBounds;

private float objectWidth;

private float objectHeight;

// Use this for initialization

void Start()

{

screenBounds = Camera.main.ScreenToWorldPoint(new Vector3(Screen.width, Screen.height, Camera.main.transform.position.z));

}

// Update is called once per frame

void LateUpdate()

{

Vector3 viewPos = transform.position;

viewPos.x = Mathf.Clamp(viewPos.x, screenBounds.x + 1.5f, screenBounds.x \* -1 - 8.5f);

viewPos.y = Mathf.Clamp(viewPos.y, screenBounds.y + 2f, screenBounds.y \* -1 - 2f);

transform.position = viewPos;

}

}

**taşlarıyoket.cs**

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class taşlarıyoket : MonoBehaviour {

void OnTriggerEnter(Collider coll)

{

if(coll.tag!="Player" && coll.tag!="baundary")

{

Destroy(coll.gameObject);

Debug.Log(coll.name);

}

//if (coll.tag == "yoltagi")

//{

// Destroy(coll.gameObject);

//}

//if (coll.tag == "yıldıztagi")

//{

// Destroy(coll.gameObject);

//}

//if(coll.tag=="mermitagi")

//{

// Destroy(coll.gameObject);

//}

}

}

**yıldızkodu.cs**

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class yıldızkodu : MonoBehaviour {

Rigidbody fizik;

public float yıldızHızı;

void Start()

{

fizik = GetComponent<Rigidbody>();

fizik.velocity = new Vector3(yıldızHızı, 0f, 0f);

}

// Update is called once per frame

void Update () {

transform.Rotate(new Vector3(0f, 1.5f, 0f));

}

}

**yolkontrol.cs**

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class yolkontrol : MonoBehaviour {

Rigidbody fizik;

public float yolHızı;

void Start () {

fizik = GetComponent<Rigidbody>();

fizik.velocity = new Vector3(yolHızı, 0f, 0f);

}

// Update is called once per frame

void Update () {

}

**köşededuranyıldızdöndür.cs**

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

using UnityEngine.SceneManagement;

public class köşededuranyıldızdöndür : MonoBehaviour {

void Start () {

}

// Update is called once per frame

void Update () {

transform.Rotate(new Vector3(0f, 1.5f, 0f));

}

}

**rotateKalp.cs**

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class rotateKalp : MonoBehaviour {

// Use this for initialization

void Start () {

}

// Update is called once per frame

void Update () {

transform.Rotate(new Vector3(0f, 1.5f, 0f));

}

}

**skyboxDöndür.cs**

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class skyboxDöndür : MonoBehaviour {

public float dönmeHızı;

void Start () {

}

// Update is called once per frame

void Update () {

RenderSettings.skybox.SetFloat("\_Rotation", Time.time \* dönmeHızı);

}

}

**oyunahakimimkodu.cs**

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

using UnityEngine.UI;

using UnityEngine.SceneManagement;

public class oyunahakimimkodu : MonoBehaviour {

public GameObject kayayolu;

public GameObject yıldız;

float taşzaman = 0f;

float yıldızzaman = 0f;

float oyunzamanıSayacı = 0f;

float taşçıkmasüresi = 0.5f; // 0.5 di bura değiştm deneme için //not==vscyn yi off yaptım her framde halinden tekli kayanın colliderını dışa aldım

int zamanım = 0;

//quality kısmında fantastikten fastest haline aldm ve fastest ın pixel light count unu 0 dan 1 yaptm

public Material[] skyboxlar;

int EskiskyboxIndis;

public Text taşçıkmasüresitext;

[HideInInspector]

public AudioSource []arkaplanMüziği;

[HideInInspector]

public int eski;

public GameObject dünyaPlayer;

//GameObject failobj;

public static int lwlnum=-1;

// public GameObject dünyaPlayer;

public GameObject []gemiPlayers;

// public GameObject dusmanGemi1;

//public GameObject denemedüşman;

float deltaTime;

GameObject music;

public GameObject failobj;

public GameObject didItObj;

bool taşGelsin = true;

void Awake()

{

arkaplanMüziği = GetComponents<AudioSource>();

music = GameObject.FindGameObjectWithTag("music");

}

void Start () { //kayanın metalyelini enable cpu instanses yaptım

//arkaplanMüziği = GetComponents<AudioSource>();

Destroy(music);

float yEksenim = Random.Range(-7.2f, 7.45f);

Instantiate(kayayolu, new Vector3(14f, yEksenim, 0f), Quaternion.Euler(0f, 0f, 90f));

// InvokeRepeating("tasZamanıAzalt", 10f, 10f);

InvokeRepeating("skyboxDegis", 30f, 30f);

InvokeRepeating("yeniarkaplanmüziğiseç", 30f, 30f);

girisArkaPlanMüziğiSeç();

girisSkyboxSeç();

//Application.targetFrameRate = 300;

// taşçıkmasüresitext.text = taşçıkmasüresi + "";

//if (lwlnum == 0)

//{

// Instantiate(dünyaPlayer, new Vector3(-10f, 0f, 0f), Quaternion.Euler(0f, 100f, 0f));

//}

//if (lwlnum == 1)

//{

// Instantiate(gemiPlayer1, new Vector3(-11f, 0f, 0f), Quaternion.Euler(0f, -90f, 90f));

//}

//if (lwlnum == 2)

//{

// Instantiate(gemiPlayer1, new Vector3(-11f, 0f, 0f), Quaternion.Euler(0f, -90f, 90f));

// Instantiate(dusmanGemi1, new Vector3(12.3f, 0f, 0f), Quaternion.Euler(0f, 90f, 90f));

//}

//Instantiate(gemiPlayer1, new Vector3(-11f, 0f, 0f), Quaternion.Euler(0f, -90f, 90f));

// Instantiate(dusmanGemi1, new Vector3(12.3f, 0f, 0f), Quaternion.Euler(0f, 90f, 90f));

if(Application.loadedLevel!=13)

{

if (PlayerPrefs.GetInt("gemiplayerİndex") == 0)

{

Instantiate(gemiPlayers[0], new Vector3(-11f, 0f, 0f), Quaternion.Euler(0f, 0f, 0f));

// Instantiate(denemedüşman, new Vector3(12.3f, 0f, 0f), Quaternion.Euler(0f, 90f, 90f));

}

if (PlayerPrefs.GetInt("gemiplayerİndex") == 1)

{

Instantiate(gemiPlayers[1], new Vector3(-11f, 0f, 0f), Quaternion.Euler(0f, 0f, 0f));

// Instantiate(denemedüşman, new Vector3(12.3f, 0f, 0f), Quaternion.Euler(0f, 90f, 90f));

}

if (PlayerPrefs.GetInt("gemiplayerİndex") == 2)

{

Instantiate(gemiPlayers[2], new Vector3(-11f, 0f, 0f), Quaternion.Euler(0f, 0f, 0f));

// Instantiate(denemedüşman, new Vector3(12.3f, 0f, 0f), Quaternion.Euler(0f, 90f, 90f));

}

}

// else if (PlayerPrefs.GetInt("gemiplayerİndex") == 1)

// {

// Instantiate(gemiPlayers[1], new Vector3(-11f, 0f, 0f), Quaternion.Euler(0f, -90f, 90f));

// }

//else if (PlayerPrefs.GetInt("gemiplayerİndex") == 0)

// {

// Instantiate(gemiPlayers[2], new Vector3(-11f, 0f, 0f), Quaternion.Euler(0f, -90f, 90f));

// }

}

void Update () {

if(taşGelsin==true)

{

taşzaman = taşzaman + Time.deltaTime;

if (taşzaman > taşçıkmasüresi)

{

float yEksenim = Random.Range(-7.2f, 7.45f);

Instantiate(kayayolu, new Vector3(15f, yEksenim, 0f), Quaternion.Euler(0f, 0f, 90f));

taşzaman = 0f;

}

}

deltaTime += (Time.deltaTime - deltaTime) \* 0.1f;

float fps = 1.0f / deltaTime;

taşçıkmasüresitext.text = Mathf.Ceil(fps).ToString();

}

void FixedUpdate()

{

yıldızzaman = yıldızzaman + Time.deltaTime;

if (yıldızzaman > 0.5f)

{

float yEksenimYıldız = Random.Range(-7.2f, 7.45f);

Instantiate(yıldız, new Vector3(20f, yEksenimYıldız, 0f), Quaternion.identity);

yıldızzaman = 0;

}

}

void tasZamanıAzalt()

{

if(taşçıkmasüresi>=0.5f)

{

taşçıkmasüresi = taşçıkmasüresi - 0.01f;

taşçıkmasüresitext.text = taşçıkmasüresi + "";

}

}

void skyboxDegis()

{

int i = Random.Range(0, 6);

if(i!=EskiskyboxIndis)

{

EskiskyboxIndis = i;

RenderSettings.skybox = skyboxlar[EskiskyboxIndis];

}

else if(i==EskiskyboxIndis)

{

int x = Random.Range(0, 6);

EskiskyboxIndis = x;

RenderSettings.skybox = skyboxlar[EskiskyboxIndis];

}

}

void girisSkyboxSeç()

{

int i = Random.Range(0, 6);

EskiskyboxIndis = i;

RenderSettings.skybox = skyboxlar[EskiskyboxIndis];

}

void girisArkaPlanMüziğiSeç()

{

int x = Random.Range(0, 9);

eski = x;

arkaplanMüziği[eski].Play();

}

void yeniarkaplanmüziğiseç()

{

arkaplanMüziği[eski].Stop();

int y = Random.Range(0, 9);

if (y != eski)

{

eski = y;

arkaplanMüziği[eski].Play();

}

else if (y == eski)

{

int a = Random.Range(0, 9);

eski = a;

arkaplanMüziği[eski].Play();

}

}

public void oyunPause()

{

SceneManager.LoadScene(0);

}

public void failÇal()

{

arkaplanMüziği[eski].Stop();

arkaplanMüziği[9].Play();

failobj.SetActive(true);

}

public void didİtÇal()

{

arkaplanMüziği[eski].Stop();

arkaplanMüziği[10].Play();

didItObj.SetActive(true);

taşGelsin = false;

}

public void yenidenOyna()

{

SceneManager.LoadScene(Application.loadedLevel);

}

}

**PlayerKodu.cs**

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

using UnityEngine.UI;

using UnityEngine.SceneManagement;

public class PlayerKodu : MonoBehaviour {

Rigidbody fizik;

//public float speed;

//public Text yıldızsayısıtext;

//public Text scoreText;

//public Text lavhalisayısıText;

public float jumpForce;

public GameObject dünyapatlama;

public GameObject kayapatlama;

public ParticleSystem partEfect;

public Material lavMat;

public Material dunyaMat;

int yıldızsayısı = 0;

int score = 0;

int astroidPatlatmahakkıSayısı =3;

AudioSource[] sesler;

float puanArttırmazamanı = 0f;

//AsyncOperation asyncLoad;

Text myScoretext;

Text myYıldızSayımText;

Text myLavhalisayısıText;

void Awake()

{

fizik = GetComponent<Rigidbody>();

sesler = GetComponents<AudioSource>();

//Invoke("loadSahne", 1f);

myScoretext = GameObject.FindGameObjectWithTag("scoreText").GetComponent<Text>();

myYıldızSayımText= GameObject.FindGameObjectWithTag("yıldızsayısıText").GetComponent<Text>();

myLavhalisayısıText= GameObject.FindGameObjectWithTag("canText").GetComponent<Text>();

}

void Start () {

//fizik = GetComponent<Rigidbody>();

//sesler = GetComponents<AudioSource>();

//asyncLoad = SceneManager.LoadSceneAsync(0);

//asyncLoad.allowSceneActivation = false;

// Invoke("loadSahne", 0.1f);

//lavhalisayısıText.text = astroidPatlatmahakkıSayısı + ""; çalışan halinde bura açık

// myScoretext.text = "ana skm";

}

// Update is called once per frame

void FixedUpdate () {

/\* float yatay = Input.GetAxisRaw("Horizontal");

float dikey = Input.GetAxisRaw("Vertical");

fizik.AddForce(new Vector3(yatay, 0.0f, dikey)\*Time.deltaTime\*speed); \*/

if(Input.GetMouseButton(0))

{

fizik.velocity = new Vector3(0f, 0f, 0f);

fizik.AddForce(new Vector3(0f, jumpForce, 0f));

sesler[0].Play();

}

transform.Rotate(new Vector3(0f, 0f, 5f));

}

void Update()

{

puanArttırmazamanı = puanArttırmazamanı + Time.deltaTime;

if(puanArttırmazamanı>0.75f)

{

score++;

// scoreText.text = "Score " + score; çalışan halinde bura var

myScoretext.text= "Score " + score;

puanArttırmazamanı = 0f;

}

//if(astroidPatlatmahakkıSayısı>0)

//{

// gameObject.GetComponent<MeshRenderer>().material = lavMat;

// partEfect.Play();

//}

//else

//{

// gameObject.GetComponent<MeshRenderer>().material = dunyaMat;

// partEfect.Stop();

//}

//lavhalisayısıText.text = astroidPatlatmahakkıSayısı + "";

}

void OnTriggerEnter(Collider coll)

{

if(coll.tag=="kayatagi" || coll.tag == "upsınır" || coll.tag == "downsınır") //alttan çıkma yaparken burda ekstra coll tag ekle

{

if(astroidPatlatmahakkıSayısı>0)

{

if(coll.tag == "kayatagi")

{

GameObject player;

sesler[2].Play();

player = Instantiate(kayapatlama, coll.transform.position, Quaternion.identity) as GameObject;

Destroy(coll.gameObject); //alt ve üst baundary i yok ediyorm bunu düzelt

Destroy(player, 2f);

astroidPatlatmahakkıSayısı--;

// lavhalisayısıText.text = astroidPatlatmahakkıSayısı + "";

myLavhalisayısıText.text = astroidPatlatmahakkıSayısı + "";

if (astroidPatlatmahakkıSayısı == 0)

{

gameObject.GetComponent<MeshRenderer>().material = dunyaMat;

partEfect.Stop();

}

}

//else if(coll.tag == "baundary")

//{

// GameObject player;

// sesler[2].Play();

// player = Instantiate(kayapatlama,transform.position, Quaternion.identity) as GameObject;

// // Destroy(coll.gameObject); //alt ve üst baundary i yok ediyorm bunu düzelt

// Destroy(player, 2f);

// astroidPatlatmahakkıSayısı--;

// //lavhalisayısıText.text = astroidPatlatmahakkıSayısı + "";

// myLavhalisayısıText.text = astroidPatlatmahakkıSayısı + "";

// if (astroidPatlatmahakkıSayısı == 0)

// {

// gameObject.GetComponent<MeshRenderer>().material = dunyaMat;

// partEfect.Stop();

// }

//}

}

else if (astroidPatlatmahakkıSayısı == 0)

{

if (coll.tag == "kayatagi")

{

Debug.Log("kayaya çarptım vay şerefsiz");

sesler[2].Play();

Instantiate(dünyapatlama, transform.position, Quaternion.identity);

//Destroy(gameObject);

// gameObject.active = false;

gameObject.SetActive(false); //burayı sonradan değiştirdim setAktive yaptım

PlayerPrefs.SetInt("yıldızSayısı1", PlayerPrefs.GetInt("yıldızSayısı1") + yıldızsayısı);

Destroy(coll.gameObject);

Invoke("Anamenuyukle", 2f);

if (astroidPatlatmahakkıSayısı == 0)

{

gameObject.GetComponent<MeshRenderer>().material = dunyaMat;

partEfect.Stop();

}

}

}

if (coll.tag == "upsınır")

{

transform.position = new Vector3(transform.position.x, GameObject.FindGameObjectWithTag("downsınır").transform.position.y+1.2f, 0f);

Debug.Log("sınıra deydim");

}

if (coll.tag == "downsınır")

{

transform.position = new Vector3(transform.position.x, GameObject.FindGameObjectWithTag("upsınır").transform.position.y-1.2f, 0f);

Debug.Log("sınıra deydim");

}

}

if(coll.tag=="yıldıztagi")

{

yıldızsayısı++;

score = score + 100;

// yıldızsayısıtext.text = yıldızsayısı+"";

// scoreText.text = "Score " + score;

myYıldızSayımText.text = yıldızsayısı + "";

myScoretext.text = "Score " + score;

Destroy(coll.gameObject);

sesler[3].Play();

if(yıldızsayısı%4==0 && yıldızsayısı!=0)

{

if(astroidPatlatmahakkıSayısı<3)

{

astroidPatlatmahakkıSayısı++;

// lavhalisayısıText.text = astroidPatlatmahakkıSayısı + "";

myLavhalisayısıText.text = astroidPatlatmahakkıSayısı + "";

gameObject.GetComponent<MeshRenderer>().material = lavMat;

partEfect.Play();

}

}

}

//if(coll.tag=="oyunsınırıtagı")

//{

// Invoke("Anamenuyukle", 2f);

//}

}

void Anamenuyukle()

{

// Debug.Log("invoke çağırıldı");

SceneManager.LoadScene(0);

//asyncLoad.allowSceneActivation = true;

}

//void loadSahne()

//{

// asyncLoad = SceneManager.LoadSceneAsync(0);

// asyncLoad.allowSceneActivation = false;

//}

}

**düşmanGemiKodu.cs**

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

using UnityEngine.UI;

using UnityEngine.SceneManagement;

public class düşmanGemiKodu : MonoBehaviour {

Rigidbody fizik;

public GameObject mermi;

public Transform mermiPos;

float zaman = 0f;

GameObject gemiPlayer1;

int darbeSayısı = 0;

//public GameObject patlamaEfekti;

GameObject oyunKontroller;

public ParticleSystem patlama;

// public ParticleSystem sonPatlama;

AudioSource[] sesler;

public Image canBarı;

float oyunbas = 0f;

//public GameObject bigEXP;

public GameObject mysprite;

Collider mycoll;

bool ptlamaEfktAçık = false;

//public GameObject didIt;

bool mermiAtsın = true;

void Awake()

{

oyunKontroller = GameObject.FindGameObjectWithTag("GameController");

}

void Start () {

fizik = GetComponent<Rigidbody>();

InvokeRepeating("mermiAt", 1f, 3f);

canBarı.fillAmount = 1f;

// gemiPlayer1 = GameObject.FindGameObjectWithTag("gemi1Sprite");

Invoke("gemiPlayerBulucu", 0.1f);

sesler = GetComponents<AudioSource>();

mycoll = GetComponent<Collider>();

}

void Update()

{

zaman = zaman + Time.deltaTime;

if (zaman > 0.5f)

{

enemiyoluyap();

}

//if (gemiPlayer1!=null) //sorun burada gemiplayer1 i bulamıor instantied edildikten sonra fix it

//{

// enemiyoluyap();

//}

// enemiyoluyap();

}

void enemiyoluyap()

{

if (transform.position.y < gemiPlayer1.transform.position.y)

{

fizik.velocity = new Vector3(0f, 0.5f, 0f);

if (gemiPlayer1.transform.position.y - transform.position.y < 0.1f)

{

fizik.velocity = new Vector3(0f, 0f, 0f);

}

}

if (transform.position.y > gemiPlayer1.transform.position.y)

{

fizik.velocity = new Vector3(0f, -0.5f, 0f);

Debug.Log("inior tamer");

if (transform.position.y - gemiPlayer1.transform.position.y < 0.1f)

{

fizik.velocity = new Vector3(0f, 0f, 0f);

}

}

}

void mermiAt()

{

if(mermiAtsın==true)

{

//Instantiate(mermi, mermiPos.position, Quaternion.Euler(0f,-90f,0f));

Instantiate(mermi, mermiPos.position, Quaternion.identity);

}

}

void OnTriggerEnter(Collider coll)

{

if(coll.tag=="mermitagi")

{

darbeSayısı++;

canBarı.fillAmount = canBarı.fillAmount - 0.05f;

//if(ptlamaEfktAçık==false)

// {

//

// ptlamaEfktAçık = true;

// }

//GameObject efekt = Instantiate(patlamaEfekti, transform.position, Quaternion.identity);

//Destroy(efekt, 2f);

patlama.Play();

Destroy(coll.gameObject);

if (canBarı.fillAmount<=0f)

{

//Destroy(gameObject);

// Instantiate(bigEXP, coll.transform.position, Quaternion.identity);

// sonPatlama.Play();

patlama.transform.localScale = new Vector3(5f, 5f, 5f);

patlama.Play();

sesler[0].Play();

//gameObject.SetActive(false);

mysprite.SetActive(false);

mycoll.enabled = false;

//transform.position = new Vector3(500, 500, 500);

// SceneManager.LoadScene(0);

//didIt.SetActive(true);

//sesler[1].Play();

mermiAtsın = false;

oyunKontroller.GetComponent<oyunahakimimkodu>().didİtÇal();

GameObject.FindGameObjectWithTag("gemi1Sprite").GetComponent<GemiPlayerKodu>().yıldızlarıEkle(); //bura ii durmadı burayı düzelt

// Invoke("AnaMenüReturn", 2.5f); //çünkü sadece tek bir gemi tagi için yapabilir

}

}

}

void gemiPlayerBulucu()

{

gemiPlayer1 = GameObject.FindGameObjectWithTag("gemi1Sprite");

}

public void AnaMenüReturn()

{

SceneManager.LoadScene(0);

}

}

**GemiPlayerKodu.cs**

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

using UnityEngine.UI;

using UnityEngine.SceneManagement;

public class GemiPlayerKodu : MonoBehaviour {

Rigidbody fizik;

// public GameObject gemipatlama;

public ParticleSystem gemipatlama;

GameObject oyunKontroller;

public Image canBarı;

// public Text canText;

int canSayısı = 3;

public GameObject mermi;

public Transform mermiCıkısYeri;

float puanArttırmazamanı = 0f;

[HideInInspector]

public int score = 0;

// public Text scoreText;

int yıldızsayısı = 0;

//public Text yıldızsayısıtext;

AudioSource[] sesler;

[HideInInspector]

public Text MycanText, yıldızsayımText, scoreumText;

//public GameObject failed;

float damageYe;

void Awake()

{

scoreumText = GameObject.FindGameObjectWithTag("scoreText").GetComponent<Text>();

yıldızsayımText = GameObject.FindGameObjectWithTag("yıldızsayısıText").GetComponent<Text>();

MycanText = GameObject.FindGameObjectWithTag("canText").GetComponent<Text>();

oyunKontroller = GameObject.FindGameObjectWithTag("GameController");

}

void Start () {

fizik = GetComponent<Rigidbody>();

sesler = GetComponents<AudioSource>();

InvokeRepeating("mermiYolla", 0.5f, 0.5f);

// canText.text = canSayısı+"";

MycanText.text = canSayısı + "";

if(PlayerPrefs.GetInt("gemiplayerİndex") == 0)

{

damageYe = 0.33f;

}

if (PlayerPrefs.GetInt("gemiplayerİndex") == 1)

{

damageYe = 0.10f;

}

if (PlayerPrefs.GetInt("gemiplayerİndex") == 2)

{

damageYe = 0.05f;

}

}

// Update is called once per frame

void Update () {

if (Input.GetKeyDown(KeyCode.W))

{

fizik.velocity = new Vector3(0f, 5f, 0f);

}

if (Input.GetKeyDown(KeyCode.S))

{

fizik.velocity = new Vector3(0f, -5f, 0f);

}

if (Input.GetKeyDown(KeyCode.A))

{

fizik.velocity = new Vector3(-5f, 0f, 0f);

}

if (Input.GetKeyDown(KeyCode.D))

{

fizik.velocity = new Vector3(5f, 0, 0f);

}

if (Input.GetKeyDown(KeyCode.Space))

{

fizik.velocity = new Vector3(0f, 0f, 0f);

}

for (int i = 0; i < Input.touchCount; i++) //ilk if in için Input.GetTouch(i).phase == TouchPhase.Moved idi

{

if (Input.GetTouch(i).phase == TouchPhase.Moved)

{

Vector2 touchPosition = Input.GetTouch(i).position;

double halfScreen = Screen.width / 2.0;

if (touchPosition.x < halfScreen)

{

//transform.Translate(Input.GetTouch(i).deltaPosition \* Time.deltaTime \* 0.35f, Space.World);

// Debug.Log(Input.GetTouch(i).deltaPosition);

//Debug.Log("sola deydim");

transform.Translate(Input.GetTouch(i).deltaPosition \* 0.0070f, Space.World);

}

else if (touchPosition.x > halfScreen)

{

// Debug.Log("sağa deydim");

//transform.Rotate(0f, 0f, Input.GetTouch(i).deltaPosition.y \* 10f \* Time.deltaTime); birleşik

transform.Rotate(0f, 0f, Input.GetTouch(i).deltaPosition.y \* 0.080f );

}

}

}

}

void FixedUpdate()

{

puanArttırmazamanı = puanArttırmazamanı + Time.deltaTime;

if (puanArttırmazamanı > 0.75f)

{

score++;

// scoreText.text = "Score " + score;

scoreumText.text = "Score " + score;

puanArttırmazamanı = 0f;

}

}

void OnTriggerEnter(Collider coll)

{

if(coll.tag == "kayatagi" || coll.tag == "baundary" || coll.tag== "dusmanMermi1")

{

Destroy(coll.gameObject);

//GameObject çarpma = Instantiate(gemipatlama, transform.position, Quaternion.identity);

//Destroy(çarpma, 1.5f);

gemipatlama.Play();

sesler[1].Play();

canSayısı--;

//canText.text = canSayısı + "";

// canBarı.fillAmount = canBarı.fillAmount - 0.33f;

canBarı.fillAmount = canBarı.fillAmount - damageYe;

//if (canSayısı>=0)

//{

// MycanText.text = canSayısı + "";

//}

//if (canSayısı<0)

//{

// //gameObject.SetActive(false);

// transform.position = new Vector3(500, 500, 500);

// // Instantiate(failed, new Vector3(0f, 0f, 0f), Quaternion.identity);

// //Invoke("levelıYenidenykle", 2f);

// PlayerPrefs.SetInt("yıldızSayısı1",PlayerPrefs.GetInt("yıldızSayısı1")+yıldızsayısı);

// oyunKontroller.GetComponent<oyunahakimimkodu>().failÇal();

//}

if (canBarı.fillAmount<=0f)

{

//gameObject.SetActive(false);

transform.position = new Vector3(500, 500, 500);

// Instantiate(failed, new Vector3(0f, 0f, 0f), Quaternion.identity);

//Invoke("levelıYenidenykle", 2f);

PlayerPrefs.SetInt("yıldızSayısı1", PlayerPrefs.GetInt("yıldızSayısı1") + yıldızsayısı);

oyunKontroller.GetComponent<oyunahakimimkodu>().failÇal();

}

}

if (coll.tag == "yıldıztagi")

{

yıldızsayısı++;

score = score + 100;

// yıldızsayısıtext.text = yıldızsayısı + "";

yıldızsayımText.text = yıldızsayısı + "";

//scoreText.text = "Score " + score;

scoreumText.text = "Score " + score;

Destroy(coll.gameObject);

sesler[0].Play();

}

}

void mermiYolla()

{

if(gameObject.tag=="gemiplayer1"||gameObject.tag=="gemi1Sprite")

{

GameObject Mermiobje = Instantiate(mermi, mermiCıkısYeri.position, transform.rotation);

Mermiobje.GetComponent<Rigidbody>().velocity = Mermiobje.transform.right \* 10;

}

}

void levelıYenidenykle()

{

SceneManager.LoadScene(0);

}

public void yıldızlarıEkle()

{

PlayerPrefs.SetInt("yıldızSayısı1", PlayerPrefs.GetInt("yıldızSayısı1") + yıldızsayısı);

}

}