Pixel's Decide

Készítette :

Pál Andor

Dali Szilárd-István

Vezető Tanár:

Dr. Szántó Zoltán

Szak :

Számítástechnika III.

Tartalomjegyzék

1. Bevezető

A videójátékok nagyon nagy népszerűségnek örvendenek napjainkban, egymás után jelennek meg az élethűbbnél élethűbb 3 dimenziós játékok, köszönhetően az informatika rohamos fejlődésének.

Már kis korunk óta fontos szerepet tölt be életünkbe a játék és ez nincs másképp a mostani gyerekekkel sem. Ők már kiskorúkban megtanulják kezelni a számítógépeket, és ezzel együtt videójátékokkal játszani is. Ezért mi úgy döntöttünk, hogy szeretnénk egy kis videójátékot készíteni.

A mi videójátékunk egy 2dimenziós pixeles játék, a neve Pixel's Decide. A továbbiakban projektünk megvalósításáról lesz szó.

2. Célkitűzések

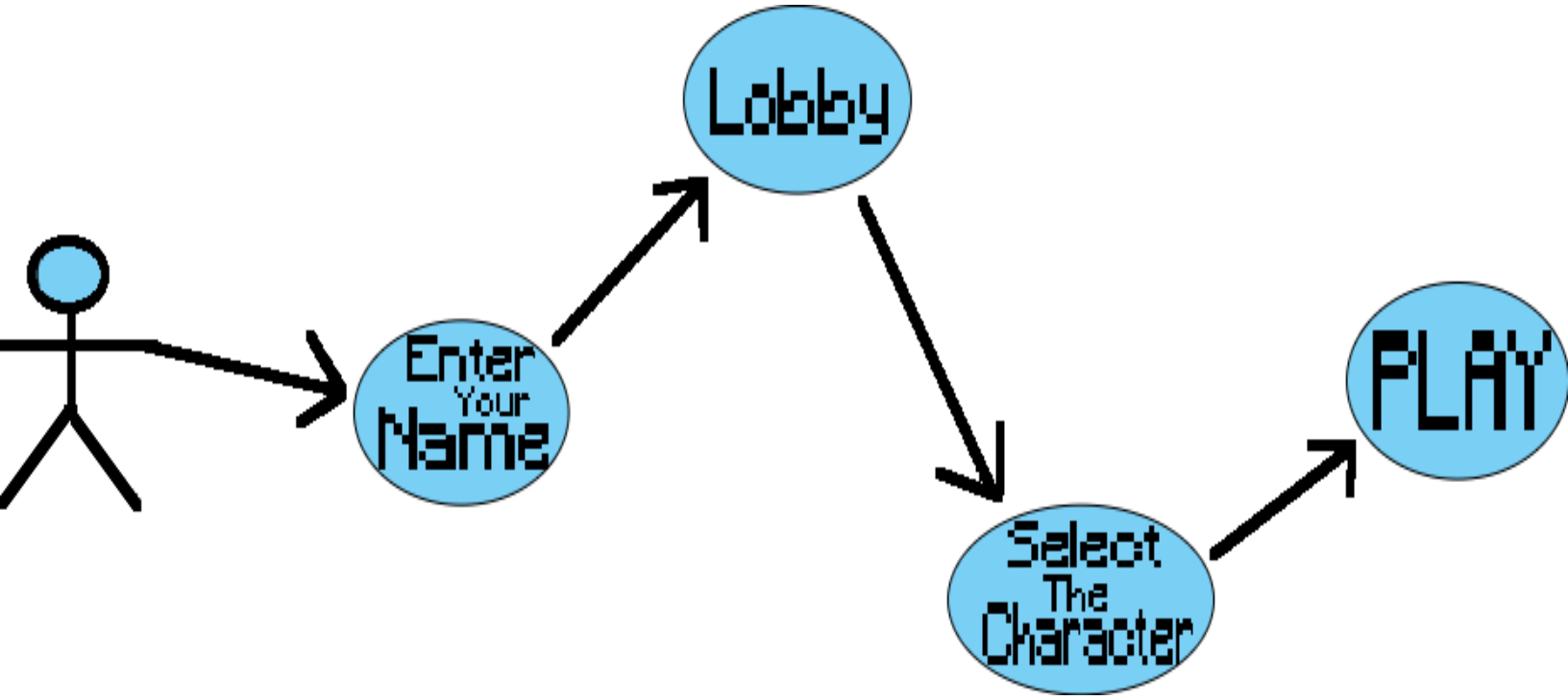
A projektünk célja megmutatni a mai fiataloknak, hogy egy pixeles videójátékkal is jól el lehet ütni az időt, továbbá azoknak, akik már játszódtak Márióval vagy más hasonló népszerű pixeles videójátékkal visszaemlékezhessenek a régi szép időkre.

Projektünk további céljai :

- A felhasználó szerveren keresztül más játékosokkal próbálhassa ki ügyességét.
- Legyen egy karakter, amit tud mozgatni (jobbra, balra futás és ugrás), valamint tud vele ütni is.
- Legyen egy akadálypálya, majd az akadálypálya végén egy aréna, ahol harcolhatnak egymás ellen a játékosok.
- A felhasználó létre tudjon hozni új szobát, de választhasson a már létrehozott szobák közül is.
- A felhasználó bármikor ki tudjon lépni a szobából
- Az a játékos aki hamarabb éri el az arénát előnyben részesüljön a többi játékossal szemben.

3. Követelmény specifikációk

3.1. Felhasználói követelmények



1. Ábra : Use-Case diagram

- Internet segítségével más felhasználók elleni játék
- Játék elindítása
- Játék használata(melyik billentyükkel lehet a karaktert mozgatni)
- Játék megszakítása bármikor
- Név megadása

3.2. Rendszerkövetelmények

3.2.1. Funkcionális

A felhasználó miután elindítja a játékot egy menü ablak jelenik meg, ahol meg kell adnia a nevét, majd a play gombra kattintva tovább léphet a lobby-ba. Fontos, hogy a felhasználónak szüksége van internetre, mivel ez egy multiplayer játék.

Ezt követően a lobby-ban találja magát a felhasználó, ahol létrhoz/belép egy szobába, ahol más játékosok is vannak.

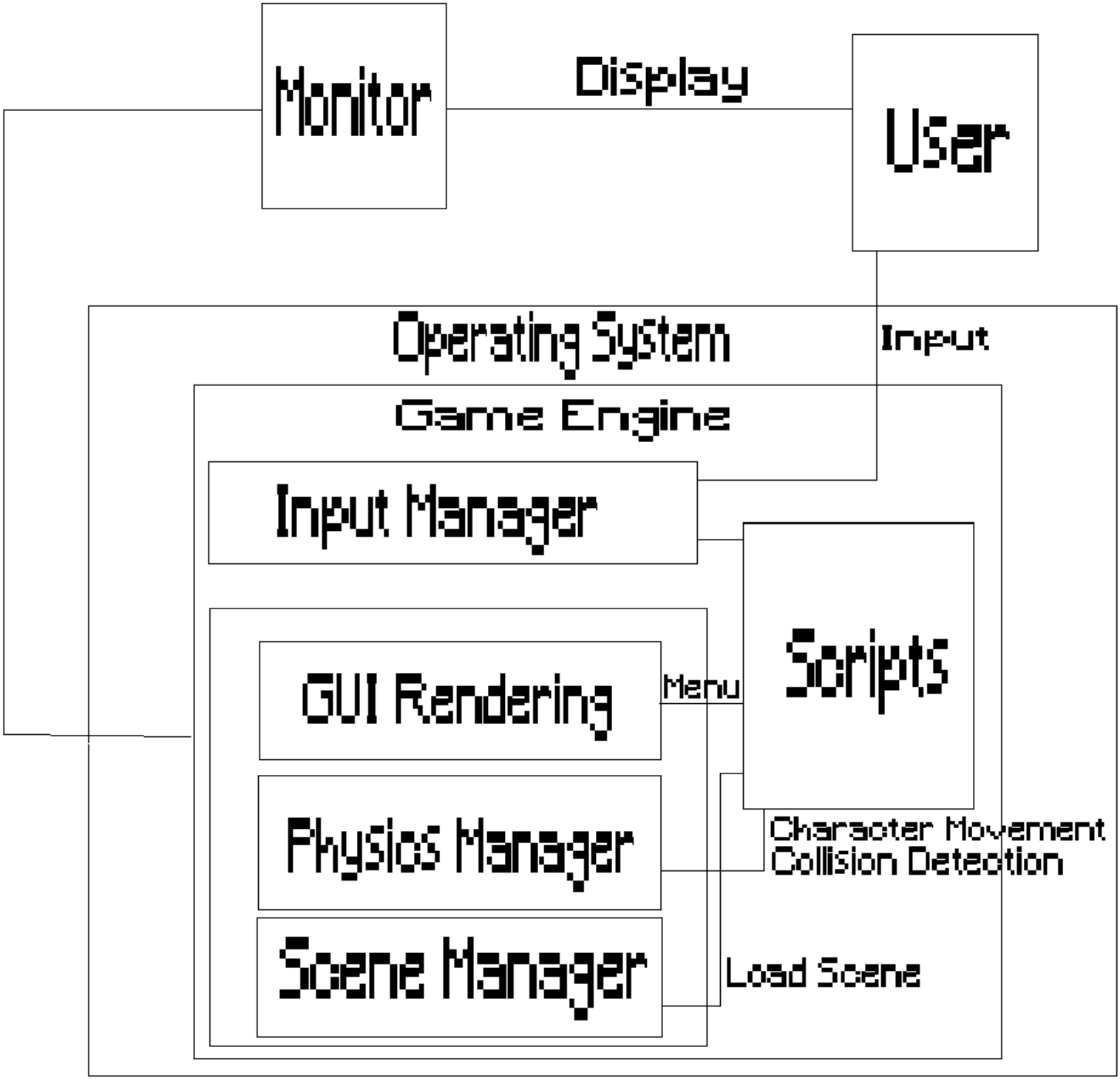
Ha mindez sikeresen megtörtént már csak egy dolog van hátra, ami nem más, mint a megfelelő karakter kiválasztása. A játék végén valakinek nyernie kell, de játékmenet közben is bármikor el lehet hagyni a szobát, persze ez az adott mérkőzés elvesztésével jár.

3.2.2. Nem-funkcionális

- Számítógép vagy laptop
- 64 bites Windows operációs rendszer
- Internet hozzáférés
- 150 MB szabad tárhely

4. Tervezés

4.1. Architektúra



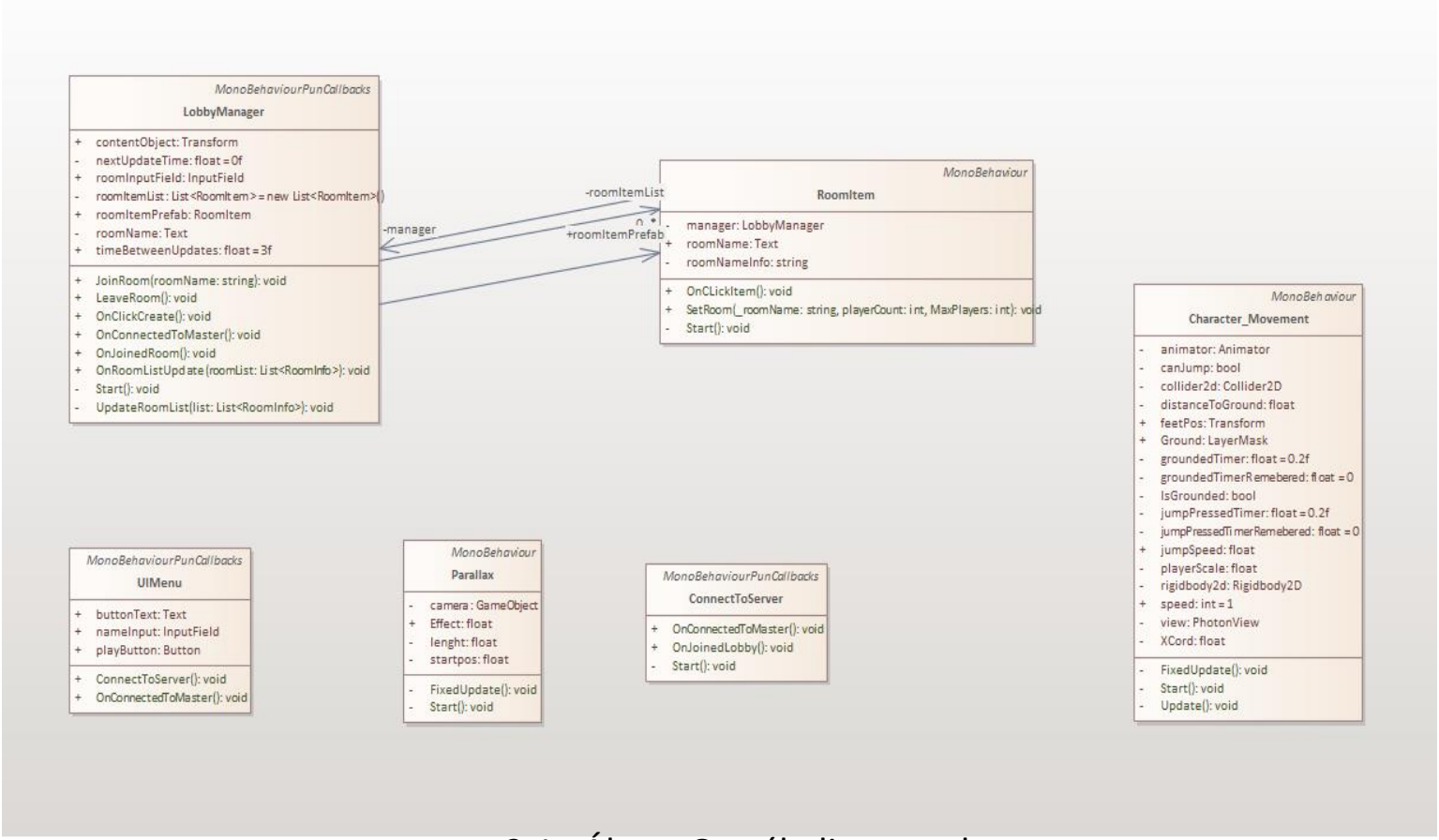
2. Ábra : Architektúra

A fenti ábrán látható a játék architektúrájának az elvi rajza. Amint látható az ábrán a felhasználó az egeret és a billentyűzetet használva különböző parancsokat adhat, ez a parancs eljut a játék bemeneteiért felelős modulba (Input Manager). A kiadott parancsok befolyással lesznek az általunk megírt kódokra (scriptek), továbbá a kódok hatással lesznek a játék kimenetére.

Amiután végrehajtódnak a kívánt parancsok, a megírt játékunk az operációs rendszert használva egy képernyőn keresztül a játék képét fogja megjeleníteni, amely másodpercenként többször is frissül.

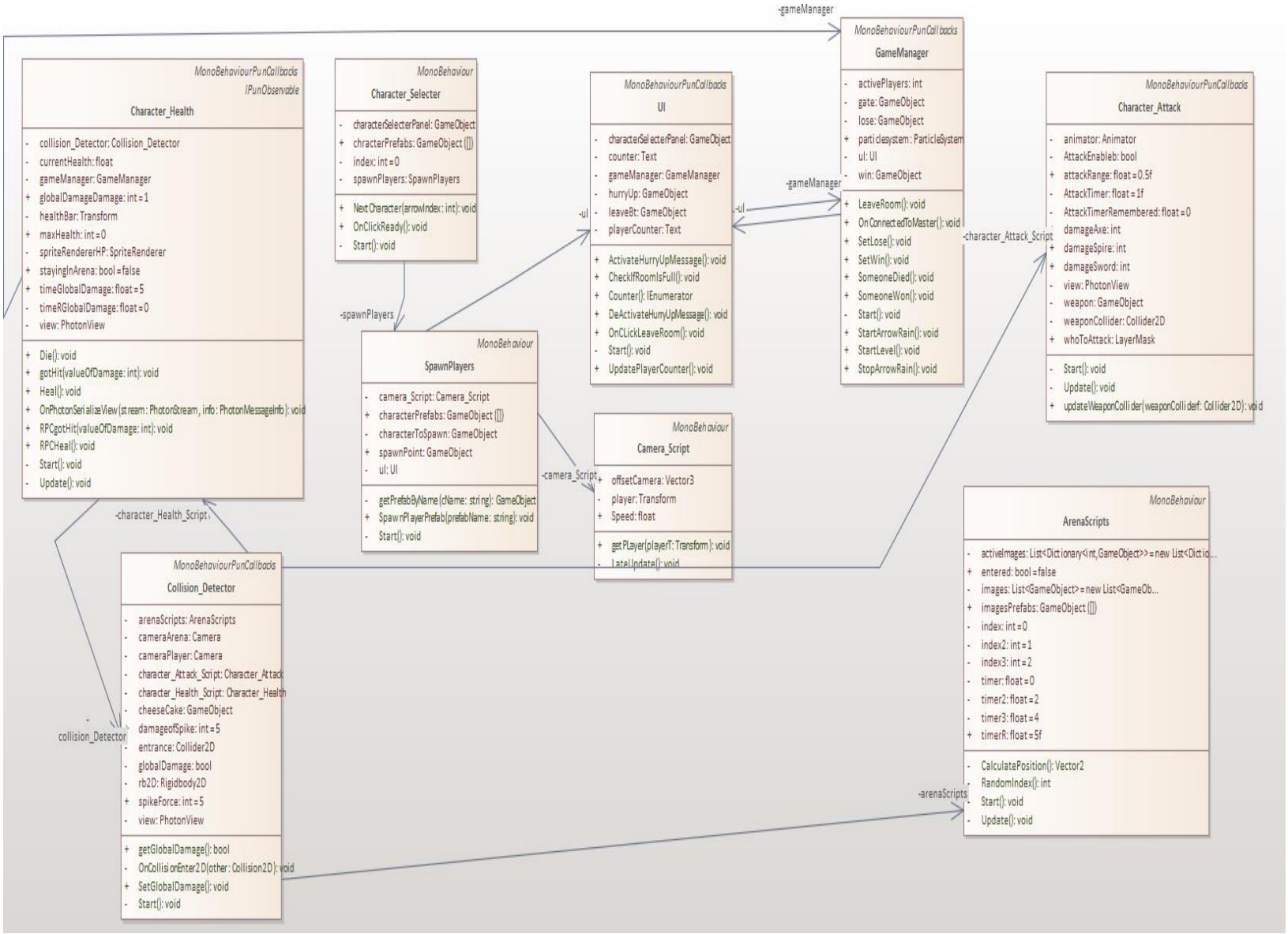
4.2. Modulok leírása

Osztálydiagramok



3.1. Ábra : Osztálydiagramok

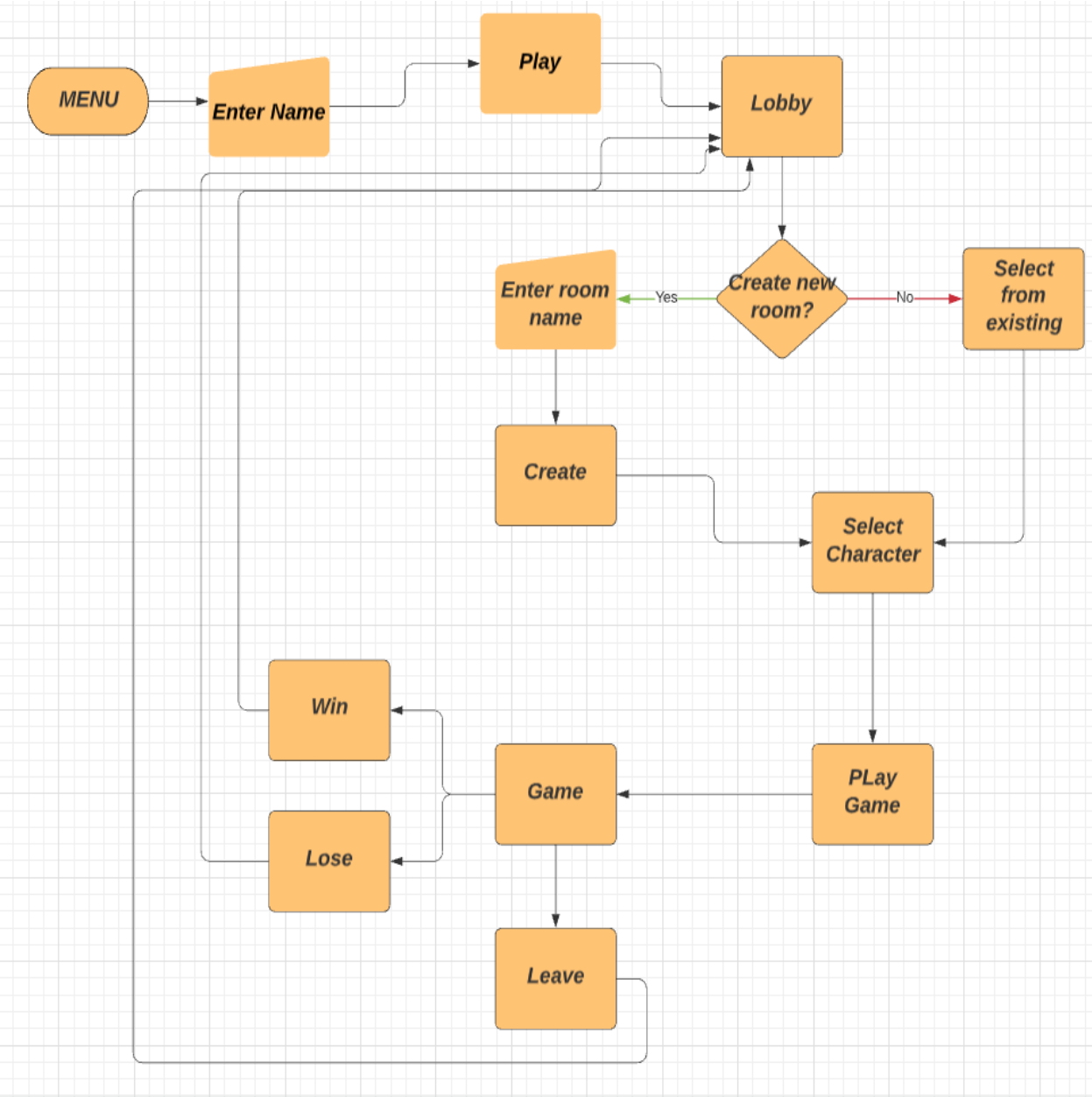
A fenti ábrán néhány script osztálydiagramját láthatjuk, pontosabban a lobby,menü és karakter mozgatásához szükséges scriptek osztálydiagramját. Megfigyelhető, hogy a legtöbb osztály a MonoBehaviour osztályból származik, amelynek van egy Start(), valamint egy Update() metódusa. Az első metódus az inicializálásért, míg a második a játék képkockáinak folyamatos frissítéséért és a megírt függvények többszöri futtatásáért felelős.



3.2. Ábra : Osztálydiagramok

A következő ábrán(3.2) további osztálydiagramokat láthatunk, ezeken is megfigyelhető, hogy nagy része a MonoBehaviour osztályból lett származtatva.

Folyamatábra(FlowChart)



4. Ábra : Folyamatábra

A **4.ábra** bemutatja, hogy hogyan is zajlik le egy játékmenet. Első lépésben a felhasználó(játékos) egy menüben beírja nevét és a play gombra kattintva megjelenik a lobby. Itt eldöntheti, hogy létrehoz egy új szobát vagy inkább választ a már meglevők közül.

Ha létrehozott egy szobát vagy belépett egy már meglevőbe, akkor már csak a karakter kiválasztása választja el a játék inditásától. Ha a játékmenet közben kilép, az azt jelenti, hogy feladta a harcot, azaz újra a lobbyban találja magát és az adott játszmát elvesztette.

Ha egy játékos beér az arénába, akkor a többiek elkezdenek sebződni, az arénába ért játékosok közül egy megeheti az ott található sütit, aminek elfogyasztásával újra maximális élettel harcolhat(erre legnagyobb esélye az első beérkező játékosnak van).

Ha valamelyik játékosnak elfogy az összes élete, akkor vesztett. Amint egy játékos marad csupán életben az megnyerte az adott futamot. Ezután a lobbyban találja újra magát.

5. Megvalósítás és működés

5.1. Tervek és munkamenet

Első lépésben eldöntöttük, hogy egy játékot szeretnénk megvalósítani. Ezután megbeszéltük, hogy **Unity**-ben fogunk fejleszteni. Felmerült a kérdés, hogy 2d-s vagy 3d-s játék legyen, végül a 2 dimenziósnál maradtunk. Ezután következett az egyik legnehezebb kérdés, mi legyen a játék témája. Arra a következtetésre jutottunk, hogy jó lenne egy kis pixeles játékot megvalósítani, de nem csak annyit szerettünk volna, hogy a karakterünk végigmegy egy akadálypályán. Ezért úgy döntöttünk, hogy egy multiplayer játékot valósítunk meg. Az elképzelés tehát az volt, hogy a játékos teljesít egy akadálypályát úgyan, de közben másik játékosok ellen verseng, majd a pálya végén az arénába érve megküzdenek egymással.

A téma és a kezdetleges elképzelés után két részre osztottuk a feladatokat, egyik rész a kódolás volt, míg a másik a design, azaz a látványvilág. A kódolásert felelős Pál Andor, és a látványvilágért pedig Dali Szilárd lett. A kódoláshoz használt technológia a **C#**, míg a látványvilágot **Aseprite**-ban valósítottuk meg.

Első lépésben megíródott a karakter mozgatásáért felelős script, amelyet egy internetről letöltött képen próbáltunk ki. Eközben design részen a föld

képkockáinak megrajzolása történt.

Miután megtörtént a tesztelés az eddig megírt, megrajzolt elemekre neki is fogtunk a többi script megírásának, valamint a többi elem megrajzolásának, és a szükséges animációk megszerkesztésének. A pálya megrajzolásában segítségünkre volt a tilemap, amivel az Aseprite-ban megrajzolt kockákkal könnyedén megszerkezthettük a kívánt akadálypályat, valamint az arénát is Unity-ben.

5. Ábra : Tile Palette

Az **5. ábrán** láthatjuk az Aseprite-ban megrajzolt kockáinkat egy “Tile Palette” – en ábrázolva. Innen kiválasztva a kockákat a Tilemap segítségével rajzolhattuk meg az akadálypályát és az arénát.

A pálya megrajzolása után a háttér megrajzolása, majd a karakterek animációja következett. A kódolás terén pedig az események leírása, karakter élete, UI , kamera, szerverrel való kommunikálás, támadás scriptek megirása valósult meg. A továbbiakban néhány részlet a scriptekből, valamint egy karakter animátora lesz látható.

```
void Update()
{

    if(view.IsMine)
    {
        jumpPressedTimerRemebered -= Time.deltaTime;
        groundedTimerRemebered = -Time.deltaTime;

        //By hitting A or D in keyboard we can move our player.
        //Checking if Input is greater than 0 or not and flipping the player model by that value.
        XCord = Input.GetAxisRaw("Horizontal") * speed;

        if(XCord != 0){
            animator.SetBool("is_running",true);

        }
        else{
            animator.SetBool("is_running",false);

        }

        if(IsGrounded = Physics2D.OverlapCircle(feetPos.position, 0.1f, Ground)){
            groundedTimerRemebered = groundedTimer;
        }

        if ( Input.GetKeyDown(KeyCode.W)){
            animator.SetBool("is_jumping",true);
            jumpPressedTimerRemebered = jumpPressedTimer;
        }
    }
}
```

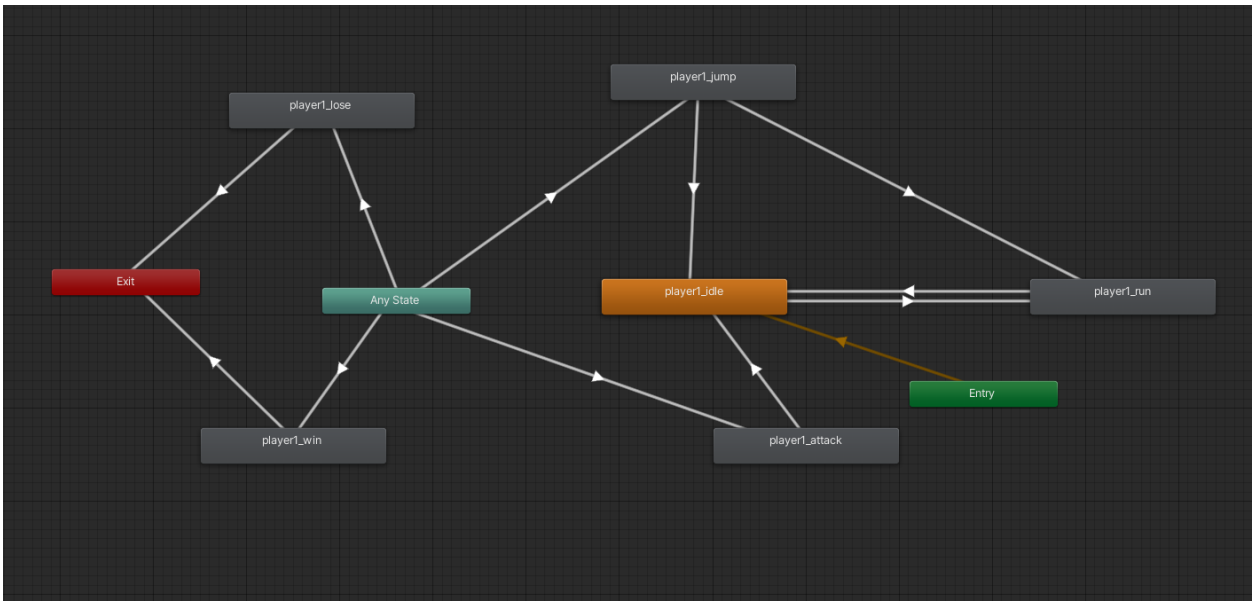
6. Ábra : részlet a Character_Movement scriptből

A 6. ábrán látható a karakter jobbra és balra mozgatásának, valamint ugrásának lekódolásából egy részlet.

```
using UnityEngine;

public class Camera_Script : MonoBehaviour
{
    Transform player;

    [Range(1,10)]
    public float Speed;
    public Vector3 offsetCamera;

    public void getPLayer(Transform playerT)
    {
        Debug.Log("Got player info :" + playerT);
        player = playerT;
    }

    void LateUpdate()
    {
        if(player != null)
        {
            Vector3 locedPosition = player.position + offsetCamera;
            Vector3 delayedPosition = Vector3.Lerp(transform.position,locedPosition,Speed*Time.deltaTime);
            delayedPosition.y = 2;
            transform.position = delayedPosition;
        }
    }


}
```

7. Ábra : Camera script

A 7. ábrán a camera scriptje látható. Itt valósul meg a karakter követése az akadálypályán.



8. Ábra : Első karakter animator

A 8. ábrán az első karakter animátora látható, a sárga téglalap, valamint a szürke téglalapok a karakter egy-egy animációja. A nyilak irányába történik az animáció lejátszása, bozinyos feltételek mellet, melyek a különböző scriptekben kapnak értéket, annak fügvényében, hogy mikor kell az adott animációt lejátszani.

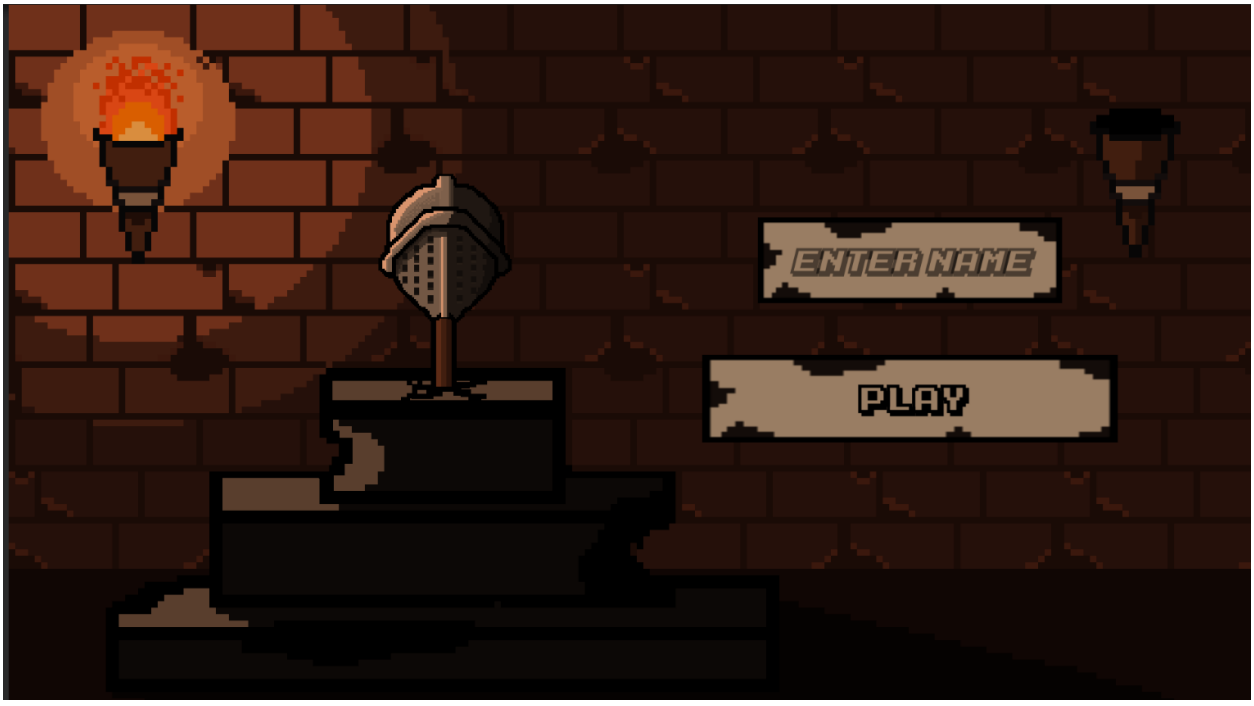
5.2. Managelés

Verziókővetésre Github-ot, valamint SourceTree-t használtunk, ez nagyon megkönnyítette a dolgunkat, mert így egyszerübb volt a közös munka, mindig meg tudtuk osztani egymással azokat a részeket, amlyek kész voltak és így a másik fél könnyen dolgozhatott velük.

Továbbá a projekt könnyed lebonyólítása végett Trello Board – ot használtunk, amelyet már az első lépésektől kezdve folyamatosan frissítettünk, még könnyebbé téve a közös munkat, valamint így jól meghatároztuk, a feladatokat, amelyeket meg kell valósítani.

6. Alkalmazás működése

A játék elindítása után a felhasználó meg kell adja a nevét majd a play gombra kattintva mehet tovább, ahogy a 9.ábrán látható.



9. Ábra : Játék menü

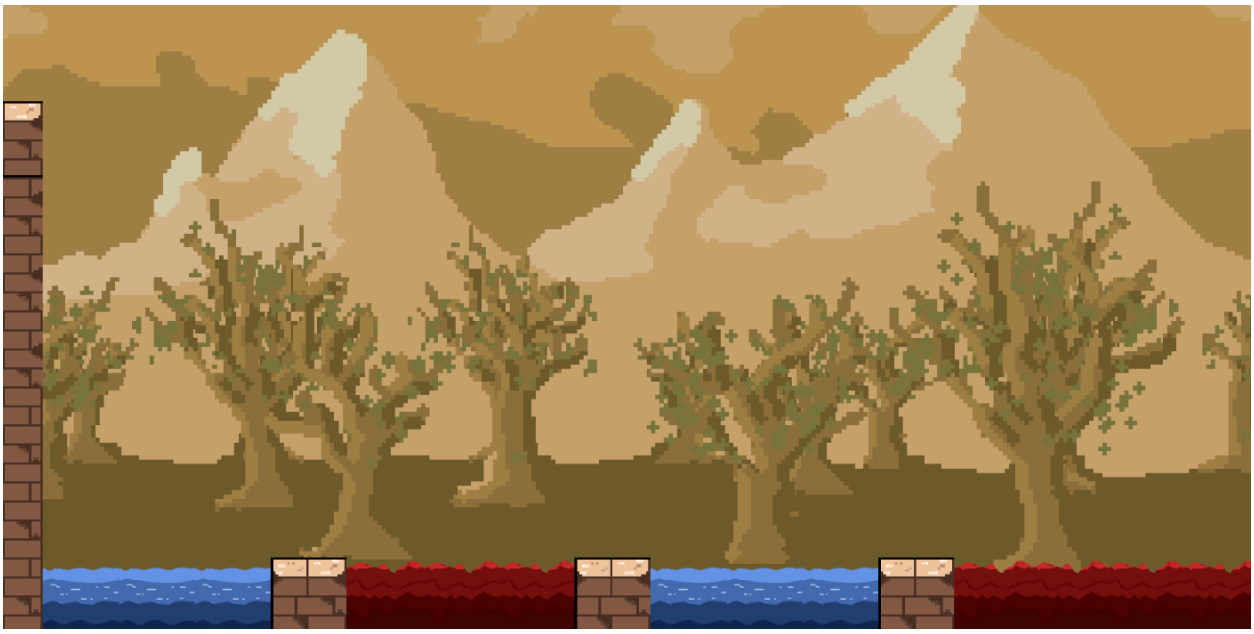
Ezt követően a lobby-ban választhat a meglevő szobák közül vagy létrehozhat egy újat, ez látható a 10. ábrán.

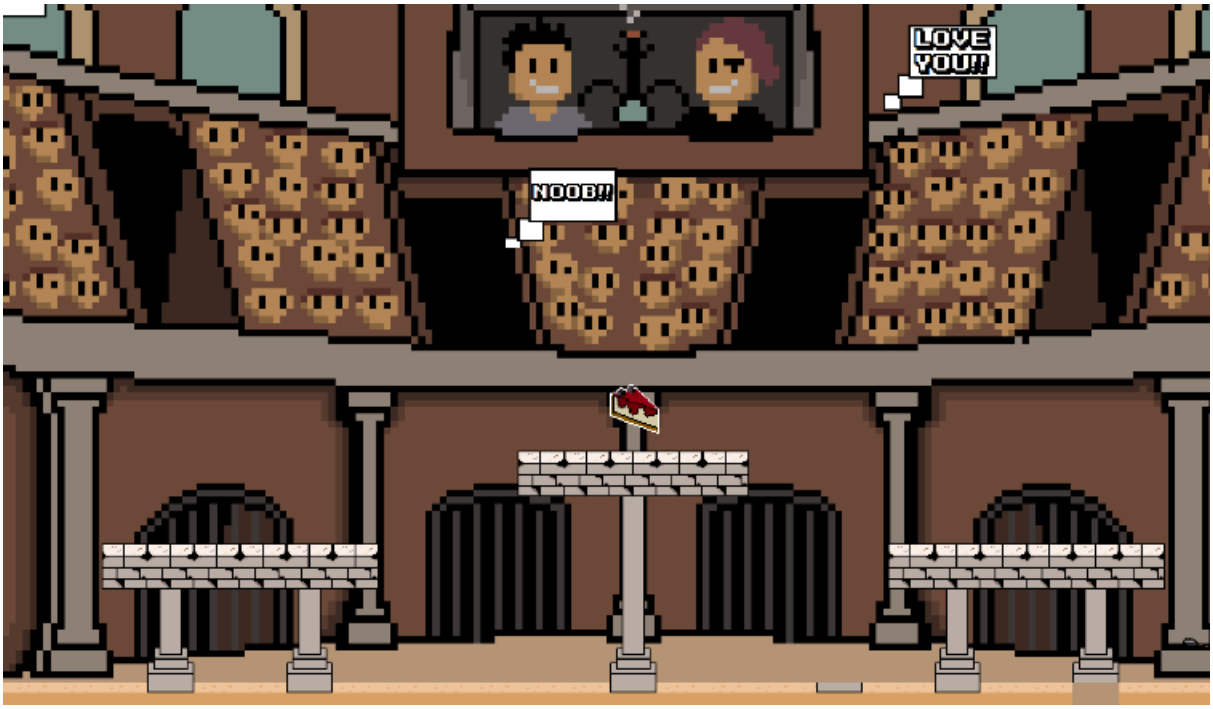
10. Ábra : Lobby

Az utolsó lépés mielőtt ténylegesen elíndul a játék a karakter kiválasztása, ez a 11. ábrán látható.

11. Ábra : Karakterválasztás

A pálya elején a falon felírat formájában megjelnik a játék célja, valamint az, hogy melyik billentyűkkel lehet mozgatni a karaktert. A továbbiakban néhány kép lesz látható az akadálypályáról és az arénáról.





7. Összegzés

Terveinket tehát sikerült nagyvonalakban megvalósítani, azonban akadtak néha hibák is, amelyek miatt újra kellet terveznünk a játékot, ilyen például az is, hogy eredetileg a fegyvereket az arénában vették volna fel a játékosok, végül a könnyebb kivitelezés miatt úgy döntöttünk, hogy végig a karakternél less a fegyver. Szerettünk Unity-ben dolgozni, mert viszonylag hamar meg lehet tanulni játékot készíteni vele, és több típusú játék is megvalósítható a felület segítségével.

7.1. További fejlesztési lehetőségek

- Jelenleg egy akadálypályát használhatnak a felhasználók, szeretnénk, ha a későbbiekben több közül is választhatnának
- Szeretnénk még több karaktert a meglevő 2 mellé
- Egy adatbázisba elmenteni a játékosok nyerési arányát,természetesen ehhez belépés és regisztráció szükséges