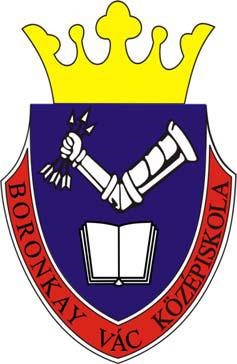
**VÁCI SZAKKÉPZÉSI CENTRUM  
BORONKAY GYÖRGY  
MŰSZAKI TECHNIKUM ÉS GIMNÁZIUM**

**SZAKDOLGOZAT**

**Foltin Milán, Papp Tamás**

**2023.**

**VÁCI SZAKKÉPZÉSI CENTRUM   
BORONKAY GYÖRGY   
MŰSZAKI TECHNIKUM ÉS GIMNÁZIUM**



**SZAKDOLGOZAT**

**Knight’s Quest**

|  |  |
| --- | --- |
| Konzulens: Wiezl Csaba | Készítette: Foltin Milán  Papp Tamás |

# **Hallgatói nyilatkozat**

Alulírottak, ezúton kijelentjük, hogy a szakdolgozat saját, önálló munkánk, és korábban még sehol nem került publikálásra.

|  |  |  |
| --- | --- | --- |
|  |  |  |
| Foltin Milán |  | Papp Tamás |

# **Konzultációs lap**

Vizsgázók neve: Foltin Milán, Papp Tamás

Szakdolgozat címe:Knight’s Quest

Program nyújtotta szolgáltatások:

* Több pályákon való játékmenet, kihívás, problémamegoldó képesség fejlesztése, észlelésfelfogó képesség fejlesztése
* Játék beállítása
* Forest pálya(legelső pálya)
* Dark forest (második pálya)
* Outside castle(harmadik pálya)
* Castle(utolsó pálya)

|  |  |  |
| --- | --- | --- |
| Sorszám | A konzultáció időpontja | A konzulens aláírása |
| 1. | 2022.10.10. |  |
| 2. | 2022.11.14. |  |
| 3. | 2022.12.12. |  |
| 4. | 2023.01.16 |  |
| 5. | 2023.02.13. |  |
| 6. | 2023.03.13 |  |

|  |  |  |
| --- | --- | --- |
| A szakdolgozat beadható: |  | A szakdolgozatot átvettem: |
| Vác, 2023. |  | Vác, 2023. |
| Konzulens |  | A szakképzést folytató intézmény felelőse |

**Tartalomjegyzék**

[**Hallgatói nyilatkozat** 3](#_Toc133216790)

[**Konzultációs lap 4**](#_Toc133216791)

[**Tartalomjegyzék 5**](#_Toc133216792)

[**Témaválasztás 6**](#_Toc133216793)

[**1 Fejlesztői dokumentáció 7**](#_Toc133216794)

[1.1 Ismertetés 7](#_Toc133216795)

[1.2 Fejlesztői környezet 7](#_Toc133216796)

[1.3 Adatszerkezet 11](#_Toc133216797)

[1.4 Használati esetmodell, szerepkörök 13](#_Toc133216798)

[1.5 Osztályok, Modulok 13](#_Toc133216799)

[1.6 Tesztelés 33](#_Toc133216800)

[1.7 Munkamegosztás 35](#_Toc133216801)

[1.8 Továbbfejlesztési lehetőségek 36](#_Toc133216802)

[**2 Felhasználói dokumentáció 38**](#_Toc133216803)

[2.1 Rendszerkövetelmények 38](#_Toc133216804)

[2.2 Alkalmazás telepítése, szükséges beállítások 38](#_Toc133216805)

[2.3 Alkalmazás használata 40](#_Toc133216806)

[2.4 Játék ismertetése 43](#_Toc133216807)

[**3 Irodalomjegyzék 47**](#_Toc133216808)

[3.1 Internetes forrásaink 47](#_Toc133216809)

[3.2 Könyvek 47](#_Toc133216810)

[**4 Mellékletek 49**](#_Toc133216811)

[**5 Összefoglalás 50**](#_Toc133216812)

# **Témaválasztás**

Mivel szeretünk videójátékokkal játszani és mindig is érdekelt minket az hogy milyen saját játékot csinálni ezért mi a játék témakört választottuk meg mint beadandó szakdolgozat, mivel úgy gondoltuk ez a legjobb döntés számunkra és szerettük volna kipróbálni milyen nehézségekkel jár egy játéknak az elkészítése, a verzió amit a projektben bemutatunk csak demó verziónak szolgál. Továbbá nem sokan vágnak bele egy játéknak az elkészítésébe mivel a legegyszerűbb játék elkészítése is tud nehézségeket okozni. A játékot Unity-ben készítettük el mivel sok oktatóanyag megtalálható róla az interneten és ezt tartottuk a lehető legjobb játékmotornak a mi játékunkhoz. Egy 2D-s platformer játékot hoztunk létre, amiben a karaktered egy lovag és a játék célja, hogy legyőzz különböző szörnyeket különböző pályákon mellette kerüld a csapdákat. A játékhoz akartunk egy külön történetet is írni viszont az nem valósult meg mivel nem volt rá időnk, és próbáltuk a játékot a lehető legélvezhetőbbre kihozni. A játék megalkotása komplex volt mivel nem egy egyszerű játékot terveztünk különböző rendszereken kellett dolgoznunk ilyenek voltak például: harcrendszer,mozgás,AI megalkotása, különböző menük. Feladataink körébe tartozott a programozás C#-ban, anyaggyűjtés, animáció készítés, mapok megalkotása(grafikus fejlesztés).

# **Fejlesztői dokumentáció**

## Ismertetés

Ezzel a játékkal bármilyen korosztály tud játszani, aki szórakozásra, kikapcsolódásra vágyik és megfelel az ízlésének. A játék ügyességi elemeket és kihívásokat tartalmaz, szörnyeket(bosst is), és változatos pályákat. A játékRPG(szerepjáték) típust képvisel 2D-s platformer megvalósítással. Csak PC-ken futtatható más eszközöket nem támogatna, viszont kisebb teljesítményű gépeket is támogat, a játék mivel nem rendelkezik nagy fájlmérettel, és komoly grafikával. Összességében ez egy izgalmas és szórakoztató játék, amely sokféle korosztály számára ajánlott.

## Fejlesztői környezet

Szakdolgozatunkhoz Unityt, UnityHubot, Microsoft Visual Studio-t azon belül is (C#-ot) használtunk.

**Unity**



A Unity egy videójáték-motor, amelyet a Unity Technologies fejleszt.A Unity segítségével háromdimenziós illetve két dimenziós videójátékokat, ezen kívül egyéb interaktív jellegű tartalmakat lehet létrehozni, például építészeti látványterveket vagy valós idejű három dimenziós animációkat.

A Unity szoftverrel való videójáték-készítés lehetséges Microsoft Windows vagy Mac OS X operációs rendszerek használatával,a játékmotor segítségével létrehozott játékok pedig futtathatók Windows, a Mac OS X, Xbox 360,PlayStation3, Wii, iPad, iPhone vagy Android alatt. 2012 júniusában pedig bejelentették, hogy aUnity a Linux rendszert is támogatni fogja. a Unitynek két főbb alkotó része van: az egyik játékok fejlesztésére és tervezésére használható szerkesztő, a másik pedig maga a videójáték-motor, amely a végleges változat kivitelezésében nyújt segítséget, ám háromdimenziós modelleket nem képes létrehozni, így azokhoz mindenképp szükséges egy 3D modellező program.

AUnity videójáték-motort megjelenése óta több díjjal is jutalmazták. 2010-ben a szoftverek kategóriájában elnyerte a Technológiai Innovációs Díjat, amelyet a The Wall Street Journal című amerikai napilap ad át minden évben. Szerintünk aUnityvel gyorsan lehet tanulni és fejlesztői környezete is felhasználóbarát mivel könnyen elérhető mappákba lehet rakni egyes kódfájlokat és különböző paneleket lehet lenyitni.[[1]](#footnote-2)

**Microsoft Visual Studio**

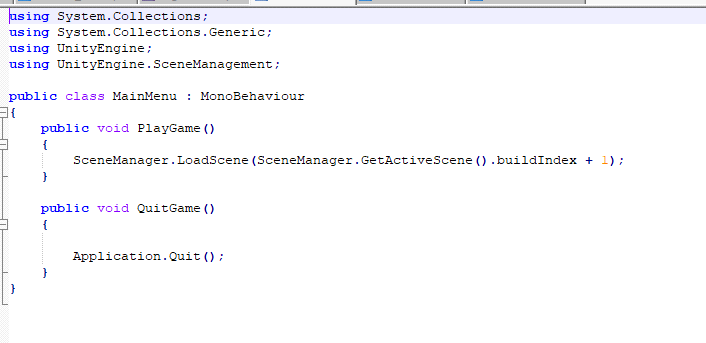


Visual Studio a Microsoft több programozási nyelvet tartalmazó fejlesztőkörnyezete, amely az évek során egyre több új programnyelvvel bővült. Jelenleg a F#, C++, C# (ejtsd: Szí-sárp) és Visual Basic programozási nyelveket, valamint az XML-t támogatja. A csomag része még a MASM (Microsoft Macro Assembler) is, ami részleges assembly támogatást biztosít.[[2]](#footnote-3)

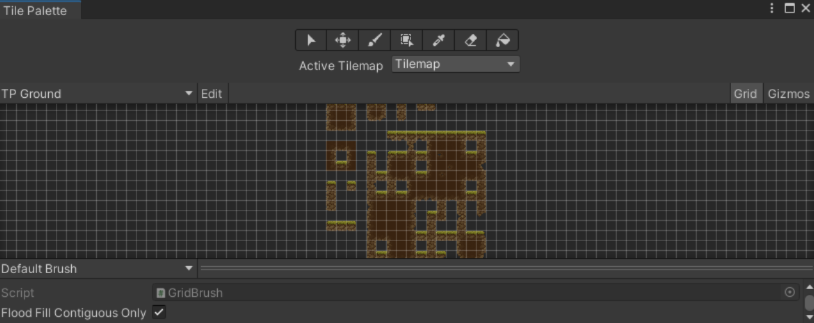
**C#**



A C# (kiejtése: szí-sárp, de ismert a cisz elnevezés is) a Microsoft által a .NET keretrendszer részeként kifejlesztett objektumorientált programozási nyelv. A nyelv alapjául a C++ és a Java szolgált. Ezzel a programozási nyelvvel próbáltunk jó kódokat írni. Egyszerű példa:



A pályák megvalósítását ennek a menünek a segítségével oldottuk meg:



Az ábrán látható TP Ground jelen esetben a kezdő pálya textúráját tartalmazza, újakat létrehozva mindig más neveket adtunk nekik és importálás előtt 32x32-es blokkokra vágtuk mivel ez volt megfelelő, ha egységes pályákat akartunk alkotni mindegyik pályának ugyanolyan felbontásban kellett lenni és a karakter felbontása is ilyen volt.[[3]](#footnote-4)

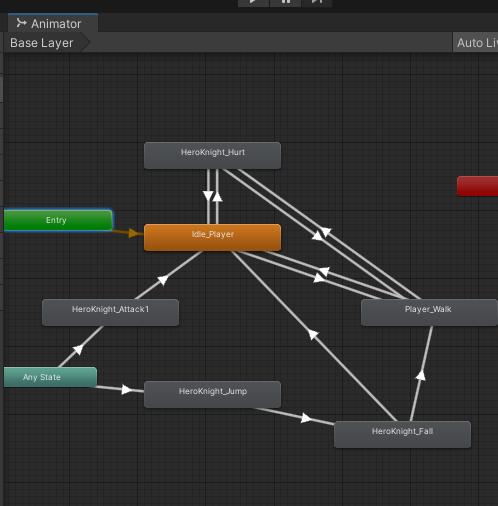
## Adatszerkezet

**Ennél a játéknál az adatszerkezet a következő képpen néz ki:**

Attól függően mihez írtuk a scriptet külön mappákat hoztunk létre egyszerű névvel, amivel könnyebb volt a fájlok megtalálása.

1. Karakteradatok: Scriptekben megírva találjuk a funkcióit és azokat egy külön mappába raktuk a megírt scripteket hozzárendeltük a hozzátartozó karakter spritehoz.
2. Ellenségadatok: Az összes ellenfél adatait szintén scriptekkel és a spriteokhoz való hozzárendeléssel valósítottuk meg.
3. Pályaadatok: Az egész pálya több különböző pálya egyesítésével valósult meg így az egész egy helyen található és közös Tilemap-et használnak.
4. Animációsadatok: Minden ellenség, és a karakter adatai a Unityben beépített Animator segítségével történt.

Példa a karakter teljes animációs szerkezetre Animator-ben:



Továbbá a megfelelő működéshez megfelelő paraméterek is szükségesek voltak például:

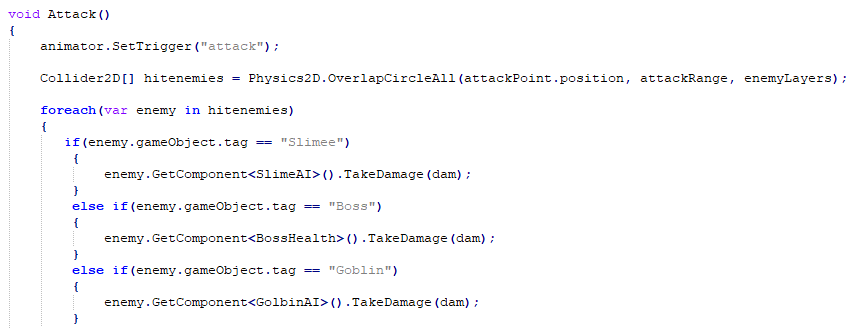


Ezek a paraméterek szükségesek voltak az animációk közötti váltáshoz és ellenőrzéshez.

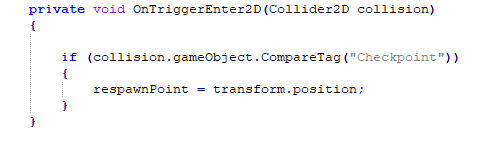
## Használati esetmodell, szerepkörök

## Osztályok, Modulok

A játék fejlesztése során rengeteg osztályt kellett létrehoznunk kezdve a játékossal, az ellenfelekkel majd a bossal a végén. A játékosnak több osztálya van, minden egyes funkció, amit képes végrehajtani külön osztályokból van meghívva(pl.: PlayerLife script). A játékos scriptjei mind külön osztályok, viszont ezek másik scriptekben nagyon kis mértékben hivatkoznak. A legtöbb másik scriptre mutató hivatkozás a PlayerAttack scriptben található ahol meghívja az egyes ellenfelek scriptjeiben található TakeDamage() metódust, ami ahogy a neve is sejteti az ellenfelekre mért sebzéshez kell, ez alapján fogy az életeröjük, a játék jelen állapotában az életerő sehol nincs jelezve így a játékosnak kell tesztelgetnie, hogy vajon hány ütést bír ki a karaktere és mennyi ütést bírnak ki az ellenfelek.

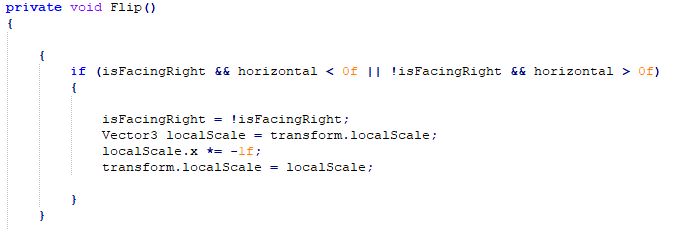


A PlayerLife scriptben található a karakter életereje valamint a checkpoint rendszer alapvető funkciója, miszerint egy bizonyos ponton áthaladva az újraéledési pont felveszi annak a bizonyos pontnak a koordinátáit így a karakter onnan fog éledni legközelebb.

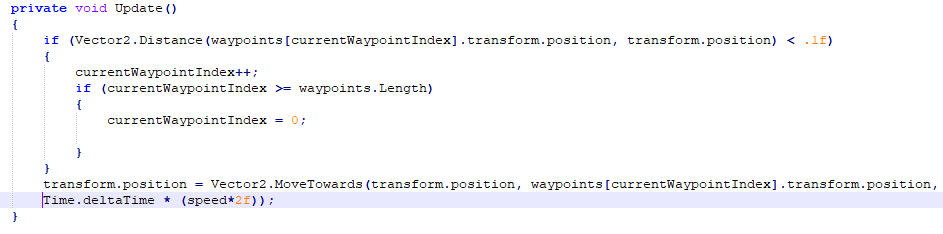


Ebben a scriptben található a csapdákkal történő interakció is, ami jelen helyzetben még csak a csapdákra való lépést tartalmazza, de a későbbiekben lehetséges, hogy beépítünk több féle csapda elemet.

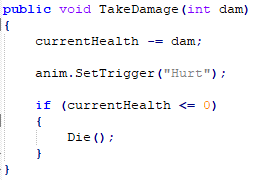
A leghosszabb script a játékosnál a mozgási script(PlayerMovement), ebben vannak meghatározva a játékos mozgási sebessége, az ugrás nagysága, az hogy merre néz a karakter és az hogy hozzá ér e a földhöz. A scriptben minden egyes résznek több tényezőnek kell megfeleljen, ahhoz hogy egy bizonyos cselekmény megvalósuljon ilyen például az ugrás vagy a flip, ahhoz hogy a karakter ugorjon teljesülnie kell az isgrounded metódusnak ami csak annyit figyel hogy a karakter a földön van e, a flip metódus határozza meg merre néz a karakter itt a játékos által végzett mozgást figyeli amerre mozgott utoljára arra néz a karakter.



Az ellenfeleknél minden egy scriptben valósul meg kivéve a waypointkövetést amihez külön script lett írva(WaypointFollower). Ez a script az aktuális távolságot figyeli az előre megadott 2 „waypoint” között, és ha ez a távolság kisebb, mint a scriptben megadott érték akkor azt a waypoint-ot elértnek tekinti és elindul a másik felé.

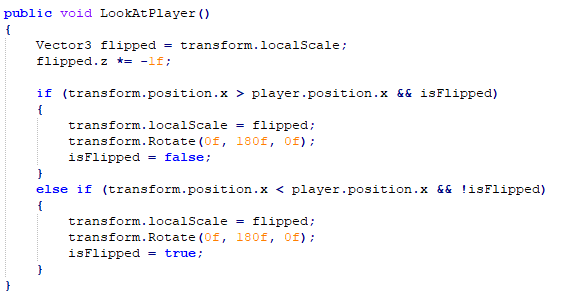


Az ellenfél scriptek elég hasonlóak egymáshoz, mert ugyanazt a működést kell megvalósítaniuk (a boss egy kivétel), ugyanúgy benne van a mozgás, az életerő, a sebzés és mindegyik ilyen scriptben benne van a feljebb említett TakeDamage() metódus.

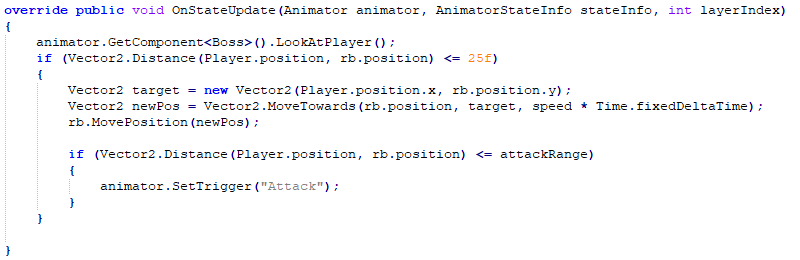


Egy scriptben ismertetem a működéseket, ez a script a Goblin működéseit tartalmazza, megtaláljuk benne a goblin életerejét, az aktuális életerejét, a waypointokat, a sebzés mértékét és a mozgási sebességet és a goblinraycast-ját(ez alapján látja a játékost). A scriptben megtalálható ugyanúgy egy flip viszont itt a waypont érzékelés alapján váltogatja azt, hogy merre néz az ellenfél. A PlayerInSight() metódus eldönti hogy a játékos benne van e a goblin támadási távolságában és ez visszatér egy bool-al ami alapján lehet meghívni a DamagePalyer() metódust ami a játékosunkat sebzi. A Die() metódus a goblin halálakor lép érvénybe, lejátssza a halál animációt és leállítja az ütközést a játékossal.

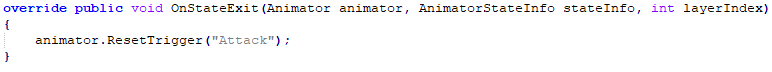
A boss script ennél bonyolultabban lett megoldva, itt már használunk az animationstate scriptet, ez belépéskor, futáskor és kilépéskor lép érvényben egyetlen state-ben. A boss 3 külön scriptből áll ezek: Boss, BossRun és BossWeapon. A Boss script a „flipelésért” felelős. Figyeli, hogy mekkora a távolság a játékoshoz képest, és ha az a boss helyzetéhez úgymond negatív vagy pozitív távolságra van megfordítja.



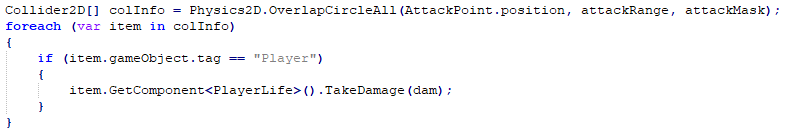
A BossRun script tartalmaz komolyabb működést. A LookAtPlayer alapján dönti, el melyik irányba kell, elinduljon az ellenség, valamint figyeli az aktuális távot itt is, ha az aktuális táv és az ellenség hatótávja egymásba vág, akkor a támadás megtörténik.



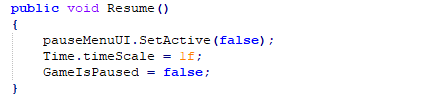
Ha ebből a távból kilép a játékos, akkor az OnStateExit lép érvénybe, ami lenullázza a képen látható triggert.



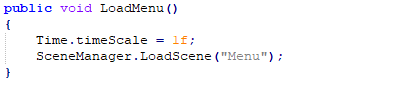
A BossWeapon script csinálja a tényleges támadást, hozzá van rendelve egy attackpoint, ami a tényleges hatótávot jelöli. Ha a játékos ebben a hatótávban marad sebzést szenved. Ezt az Attack() metódus intézi. Eldönti, hogy a támadás milyen collider-eket ért el, és ha ez megegyezik a játékos collider-ével akkor a játékos sebződik.



A játék ezen része tartalmaz még egy Menu scriptet, ami azt figyeli hogy a játékos lenyomta e az escape gombot és ha lenyomta akkor megállítja a játékot, feldobja a menüt és a játékos választhat az alábbi funkciók közül: Resume, Menu és Quit. A resume csak folytatja a játékot,



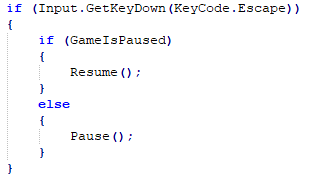
Menü gomb kattintására visszavisz a főmenübe:



A quit gomb kattintására a játék bezárul:

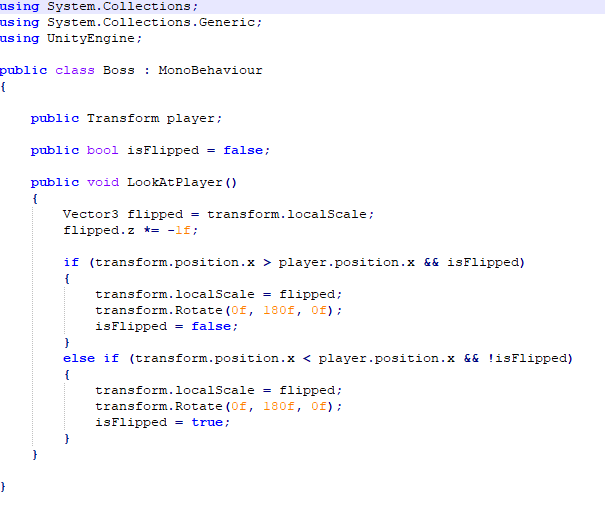


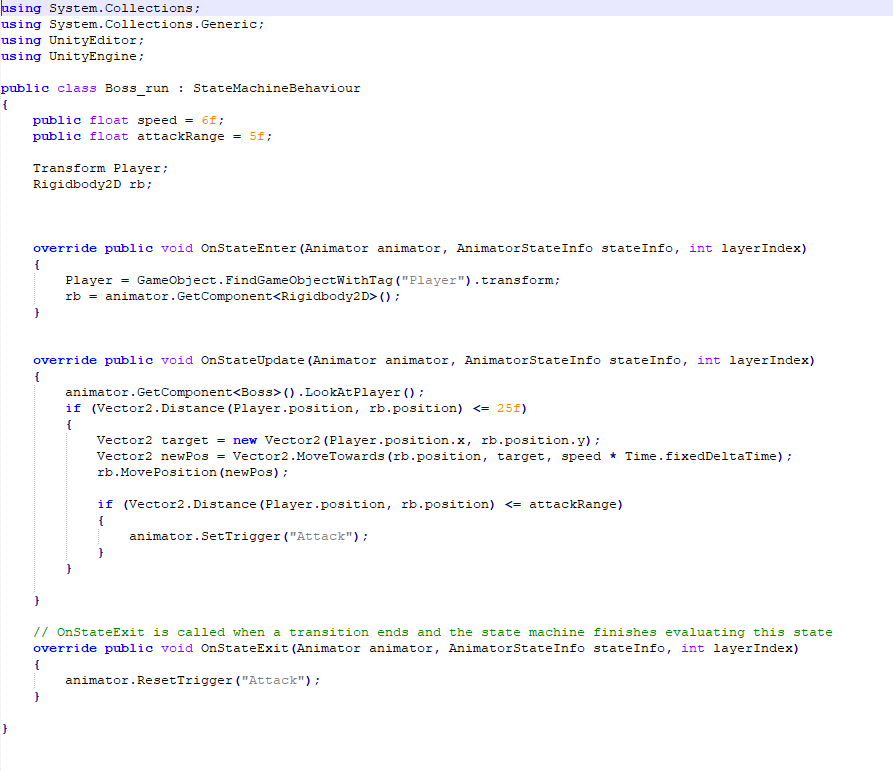
Gomb lenyomást figyelő script:

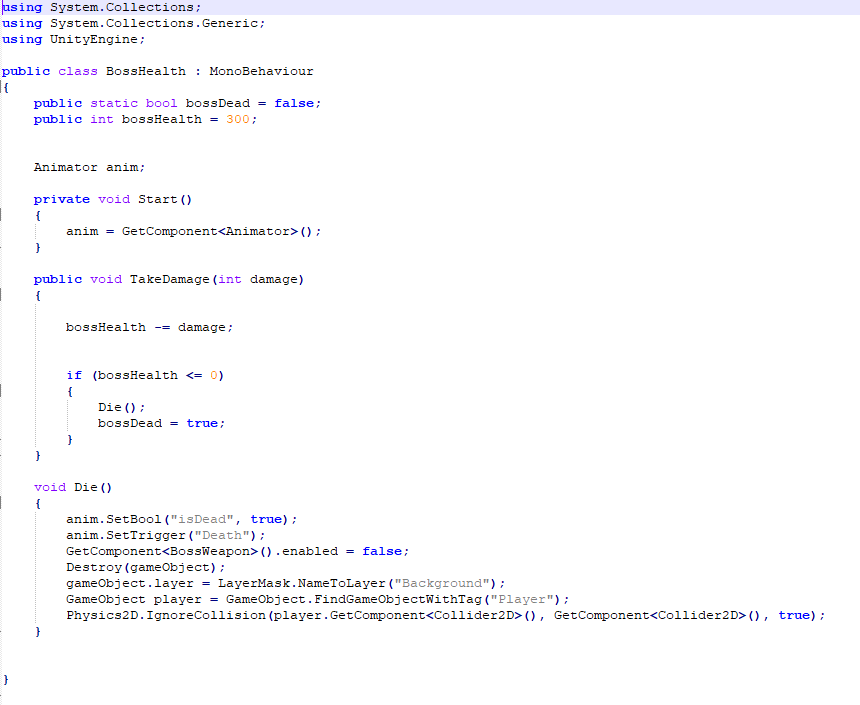


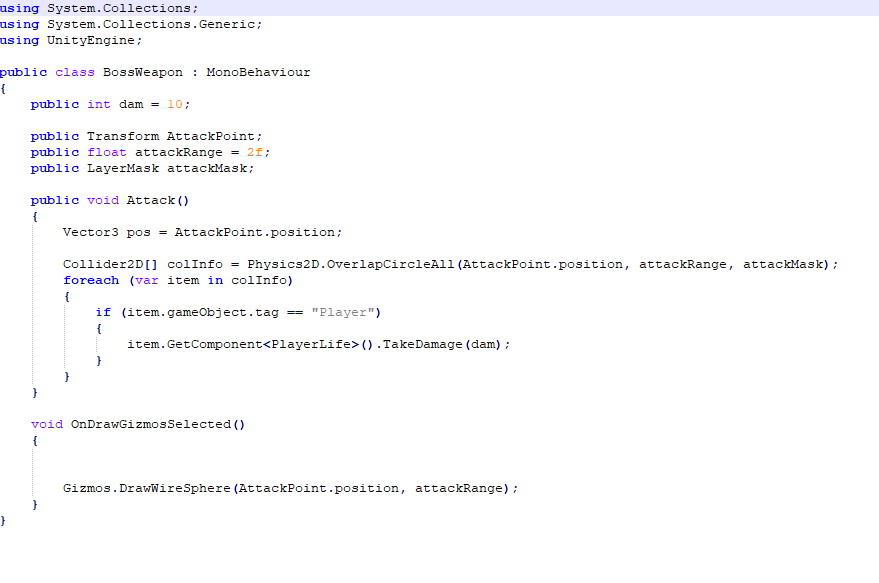
**Osztályok:**

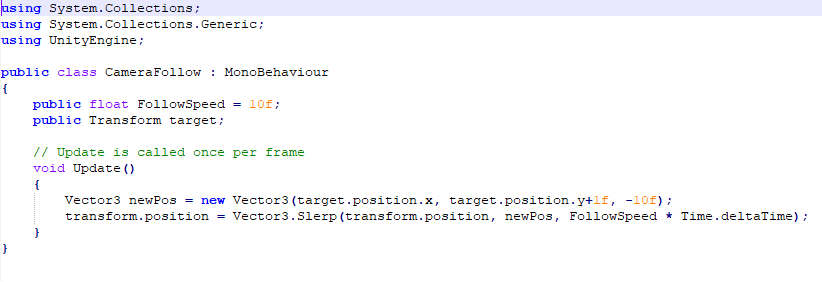
Boss.cs:



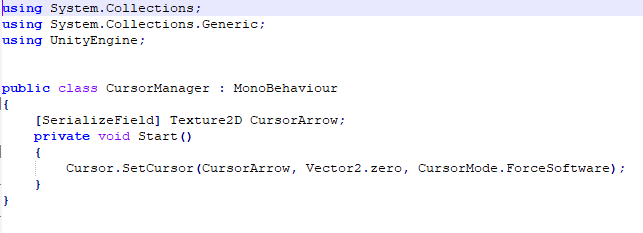
Bossrun.cs:

Bosshealth.cs:

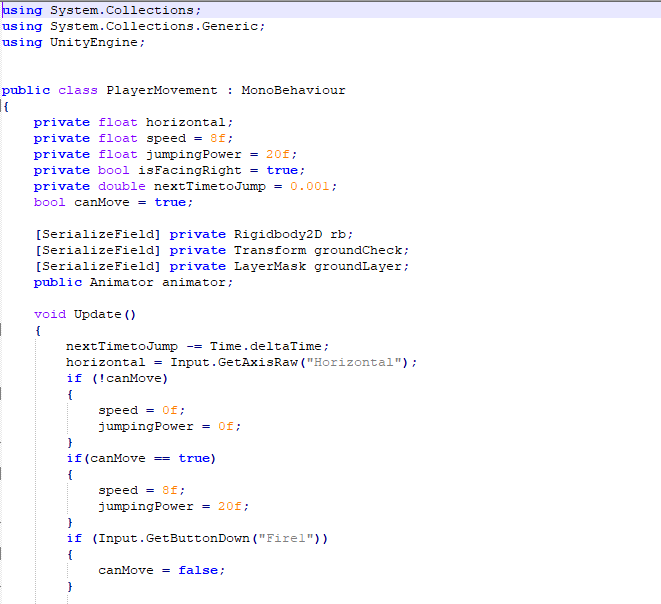
Bossweapon.cs

Camerafollow.cs:

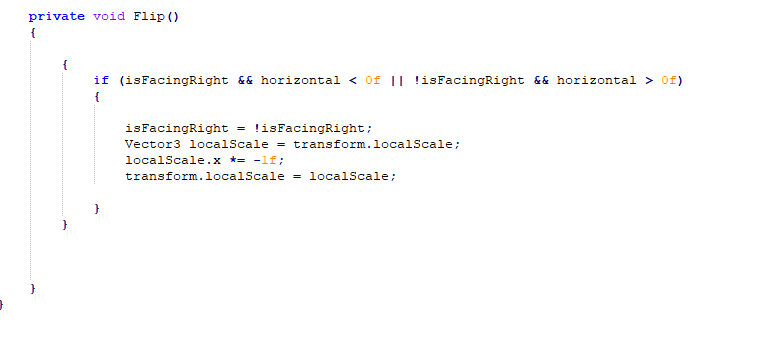
Cursormanager.cs:



Playermovement.cs:

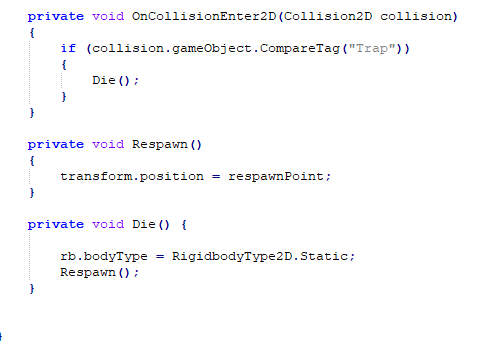


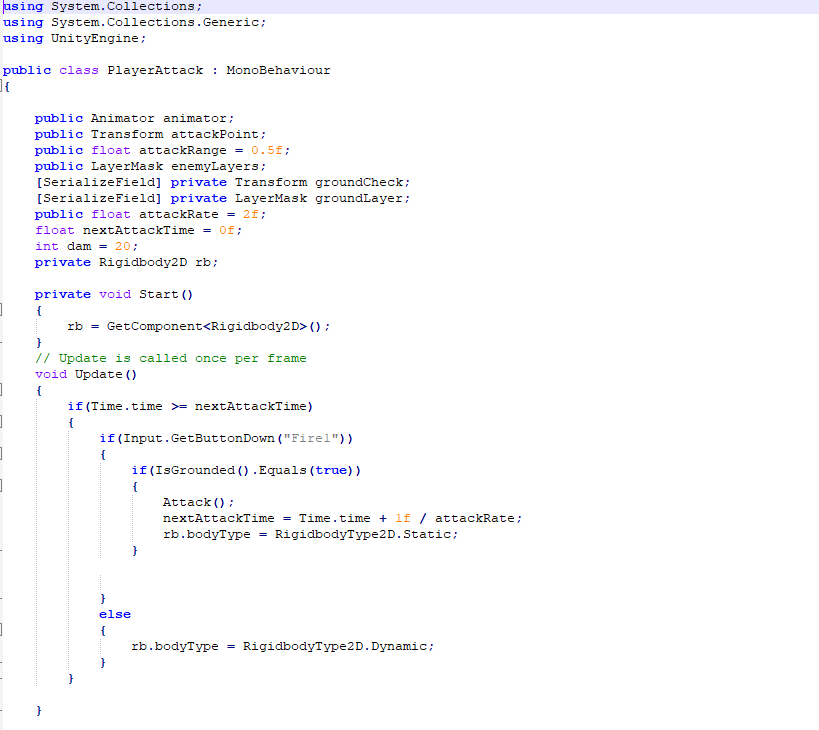
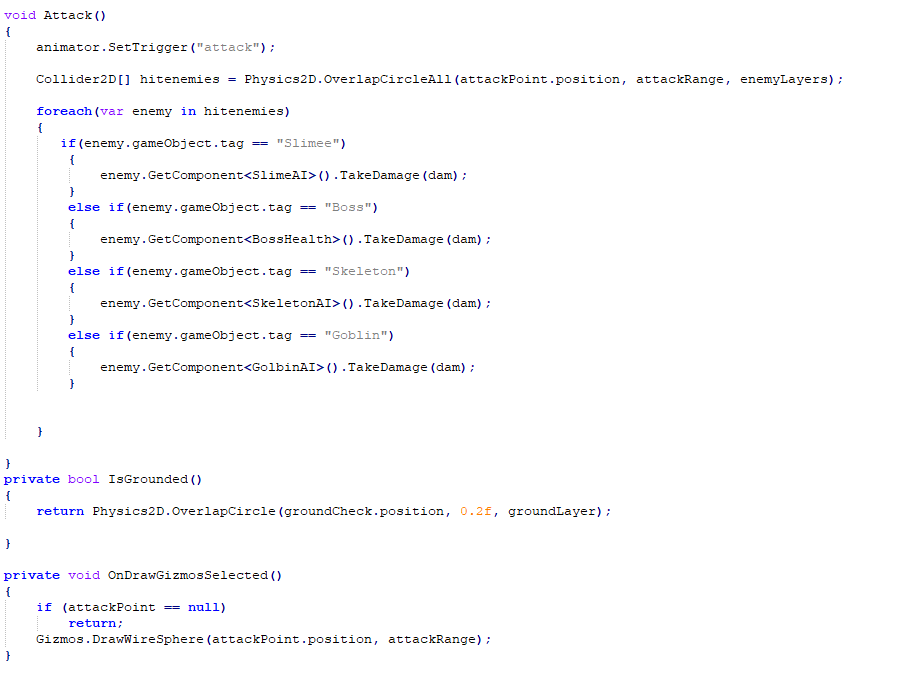




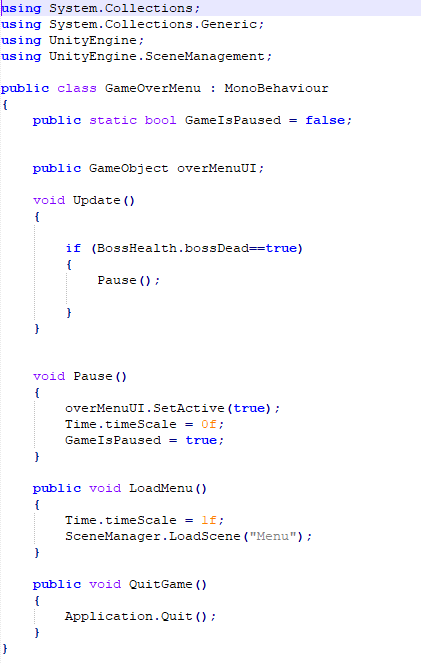
Playerlife.cs:



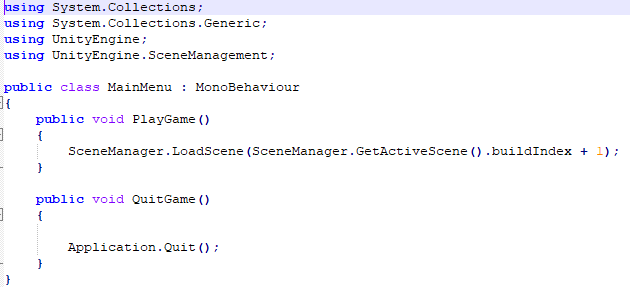


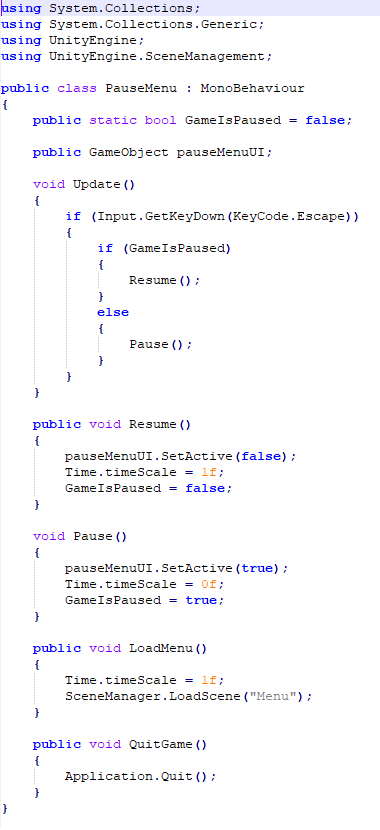
Playerattack.cs:

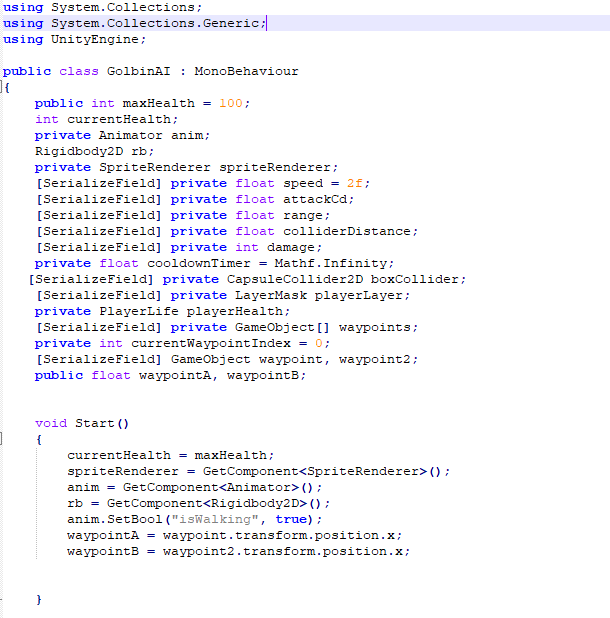
**Menük:**

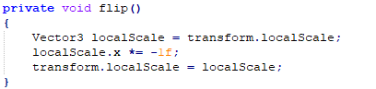
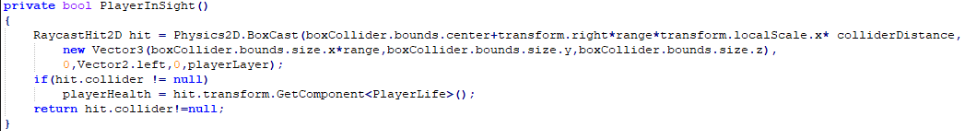
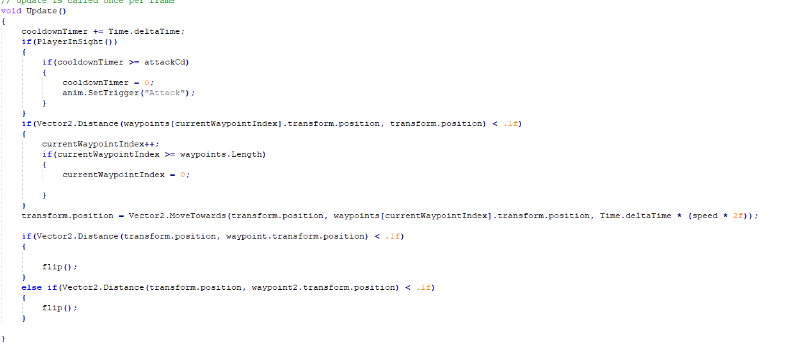
Gameovermenu.cs:

Mainmenu.cs:

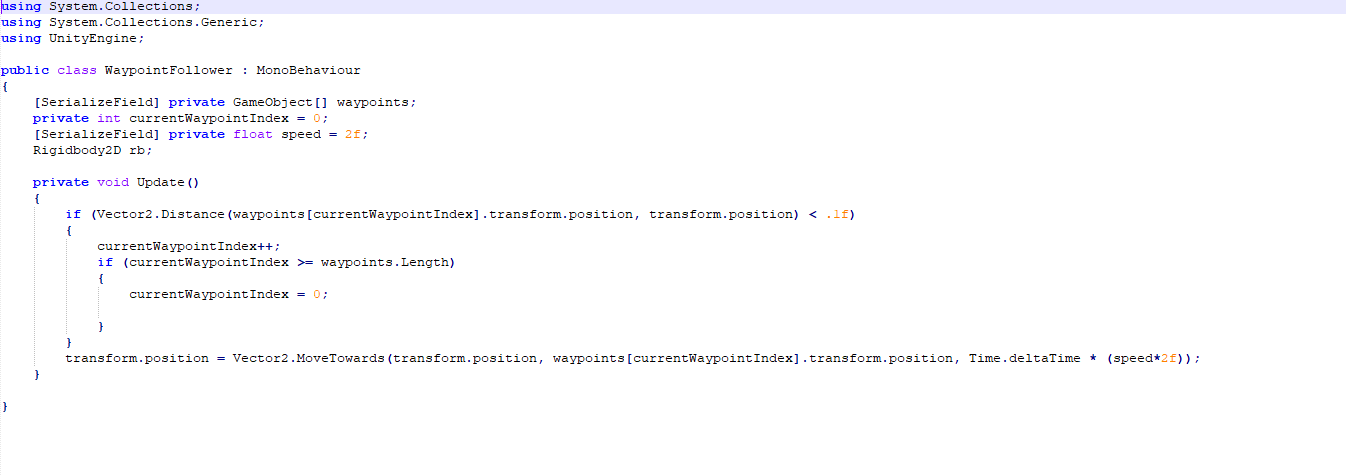


Pausemenu.cs:

Goblinai.cs:



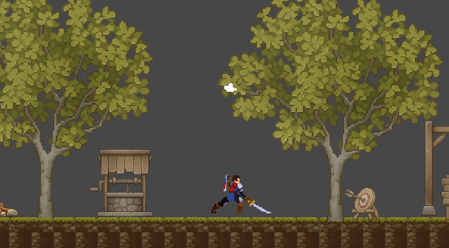


Waypointfollower.cs:

## Tesztelés

A játékot a fejlesztés során folyamatosan teszteltük. Rengeteg hibába futottunk a mozgásnál és a támadásnál, ezeknek a hibáknak diagnosztizálása és kijavítása rengeteg időt vett igénybe. Ilyenek voltak még továbbá például: a pályatesztelés,menüvezérlés helyes működése. Az elején nagyon nehezen készítettünk pályákat és csapdákat, sok oktató videót néztük arról is hogyan lehetne minél jobb egy pálya és próbáltunk olyan elemeket belerakni, amik élvezhetőbbé teszi a felhasználónak a játékot. A layerek és különböző átfedések felismerésével nagyban javult az egyes pályáknak kinézete ezek a játék végigjátszása során tapasztalható is. Tehát az első legnagyobb nehézséget a pályák okozták és annak kitalálása és összefűzése, a második legnagyobb pedig a funkciók voltak. Általában úgy teszteltünk, amikor megcsináltunk egy adott pályát vagy script funkciót utána megnéztük, hogy működik-e és ha nem akkor javítottunk, illetve ha később is felismertünk egy helyesebb működést, akkor át írtuk vagy módosítottuk. A tesztelési folyamatokat folyamatosan megbeszéltük és egyeztettünk Team-sen vagy Discordon keresztül. A folyamatos kommunikációink miatt jól tudtunk dolgozni vagy hibát javítani bármilyen probléma esetén. Viszont még így is vannak olyan részek a játékba, amik még fejleszthetőké lennének és sokkal jobb működése is lehetne ilyen például: a támadás,mozgás,AI viszont ezek sokkal több időt igényeltek volna és mivel nem volt időnk inkább a meglévő helyes funkciókat próbáltuk javítani.

Képi példa a támadás teszteléshez:



Illetve példa a pályához mennyivel lett jobb a fent láthatóábránál:



Kisebb nehézségek voltak még a colliderek beállítása a karakternél hogy ne akadjon bele semmilyen esetbe se a földön. Karaktereknél capsulecollider-t használtunk, míg a föld és csapdákon boxcollider-t ez azért volt, hogy a két collider ne tudjon úgy érintkezni, hogy beakadjon. Más kisebb nehézség volt még a helyes kamera követés annak a megfelelő sebesség megadás, és hogy folyamatosan a karakteren legyen.Legkönnyebb tesztelések közé tartozott a mozgás és a menüvezérlés tesztelése.

Összeségében rengeteget kellett foglalkozunk a tesztelés részévelis,hiszen egy játékbanelengedhetetlen tesztelni egy meglévő funkciót vagy módosítást.

A gép specifikációi, amin teszteltük:

CPU: Intel®Core™I3-6100

Memória(RAM):16 GB

GPU: Intel®HDGraphics 530

## Munkamegosztás

**Foltin Milán:**

* Menük és vezérlésük
* Pálya design
* Karakter mozgás debugolása
* AI animációik és vezérlésük részben
* Mozgóplatform
* Boss támadás fixálása
* Karakter támadás fixálása

**Papp Tamás:**

* Checkpoint rendszer
* Karakter mozgás alapjai
* Karakter támadás alapjai
* Boss scriptek,animációk
* AI animációik és vezérlésük részben
* Halál logikája

Ami nincs a felsoroltak között az közös munka volt, folyamatos kommunikáció mellett. Debugolásokat és ötleteléseket mindig egyeztetve oldottuk meg. Discordon és Teamsen valósítottuk meg nagyrészét illetve iskolában személyesen. A projekten otthon és iskolában is állandóan dolgoztunk.

## Továbbfejlesztési lehetőségek

Rengeteg ötletünk volt még hogy mit lehetett volna hozni még a játékba, hogy jobb legyen. Ilyennek gondoltuk volna még a játék elején a karakter kosztümizálását, az életjelzését a karakternek, több mapok, teleportálás másik szintre, más bossok, komplexebb harcrendszer és jobb mozgás, a játéknak a története, feladatok teljesítése, gyűjthető dolgok, inventory, ládák nyitása és szerzése,hang.

1. **Karakter mozgása:** A játékba szeretnénk több kihívást jelentő részt belerakni, ehhez véleményünk szerint elengedhetetlen a karakter megfelelő mozgásának megalkotása pl.: dupla ugrás, sprintelés, guggolás, gurulás.
2. **Életerök megjelenítése:** A játékos és az ellenfelek aktuális életeröjének megjelenítése, hogy növeljük a játékélményt és a játékos is tisztában legyen, hogy jelenleg mennyi életeröje van.
3. **Pályák:** Jelenlegi verzióban egy pályára van sűrítve az egész viszont a jövőben szeretnénk több pályát is létrehozni, hogy változatosabb legyen a játék.
4. **Bossok:** A pályák végére terveztünk többféle boss-t berakni, és egyre nehezebb lenne őket megölni a játék haladása során.
5. **Harcrendszer:** A jelenlegi harcrendszer kifejezetten egyszerű, szeretnénk további harci elemeket behozni pl.: kitérés, varázslatok, különféle távolsági támadások, összetett kombinált támadások.
6. **Történet:** A játék mostani verziójának egyetlen célja hogy végig juss a pályán, szeretnénk ezt megváltoztatni egy érdekes történet behozatalával.
7. **Quest rendszer:** A pályákon különböző NPC-k adnának küldetéseket, amit a játékos valamilyen jutalom ellenében teljesíthet. Ezeket egy dialógussal ismertetnék az NPC-k. Lenne kötelező a történetbeli haladás érdekében.
8. **Tárgyak:** Többféle fegyver és páncél behozatala, ezek különböző szinteken lennének elérhetők és kihatással lennének a karakter életeröjére, sebzésére, elszenvedett sebzésére. Ezek cserélhetők lennének.
9. **Szintrendszer:** Tapasztalati pontokat kapnál a legyőzött ellenfeleknél és a megtalált titkos helyekért és a történethaladásért. Egy szintlépés után növelhetőek lennének a státuszpontok.
10. **Státusz:** Egy olyan  megjeleníthető panel lenne, ahol a karaktered státuszpontjait láthatnád és eloszthatnád olyan statisztikákra, amik erősítik a karaktered, valamint ezektől függene, milyen felszereléseket tudna használni a karakter.
11. **Inventory:** Ez egy olyan megjeleníthető panel ahol az összegyűjtött tárgyakat, fegyvereket, páncélokat jelenítené meg és tulajdonságainak kiírása érdekében hoznánk be az inventory-t(tárgykészletet). Innen lehetne eladni bármit, vagy vásárolni.
12. **Boltrendszer:** Ez egy olyan rendszer lenne ahol egy boltos NPC karakterrel tudnál kereskedni, akinél különböző árukat tudnál vásárolni, vagy eladni.
13. **Mellék történetek:** A különböző NPC-k történetét lehetne kideríteni a küldetéseik során felvehető rejtett tárgyakból.
14. **Ellenfelek:** Különböző pályákhoz különféle ellenfelek behozatala, szintén erősödve a szintnek megfelelően, mint a bossoknál, ez javítaná a játékélményt és növelné a kihívást.
15. **Speciális effektek:** Speciális effektek hozzáadása az animációkhoz, például robbanás, szörnyek sebzésénél vérzés.
16. **Minijátékok:** Minijátékok amik, szórakoztatóak lennének, vagy a történethez kapcsolódnának.
17. **Hangok:**Főmenünél egy alap háttérzenét terveztünk volna, továbbá a játékon belüli is akartunk a harcrendszernél, illetve minijátékoknál.
18. **Frissítések:** A játékot úgy akartuk volna tovább fejleszteni, hogy folyamatos frissítéseket hoznánk be így mindig egyes részek, újak lennének.

# **Felhasználói dokumentáció**

## Rendszerkövetelmények

Bármilyen asztali számítógépen működik és csak Windows operációs rendszeren futtattható, nem támogatja se a Mac se az Android operációs rendszereket. Integrált videókártyával már élvezhető szinten futtattható nem követel ennél nagyobb gépigényt.

Példa milyen minimális rendszerigény szükséges hozzá:

CPU: Intel®Core™I3-6100

Memória (RAM):16 GB

GPU: Intel®HDGraphics 530

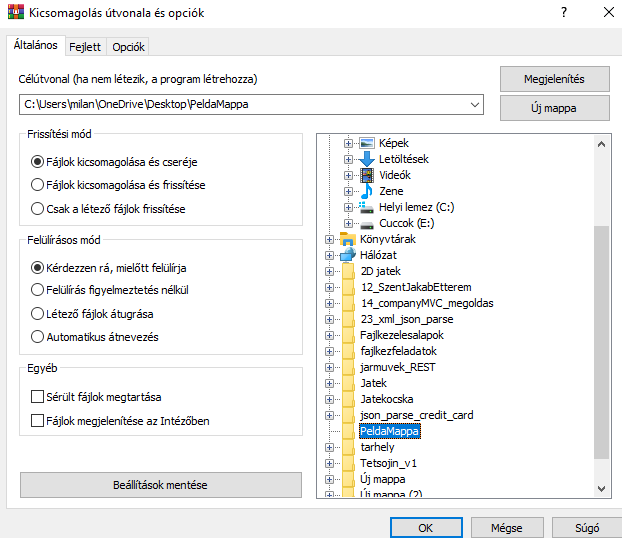
Internet nem szükséges a futtatthatásához.

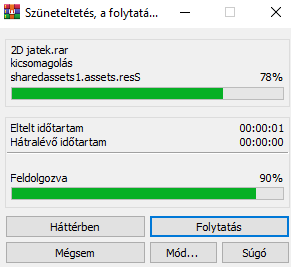
## Alkalmazás telepítése, szükséges beállítások

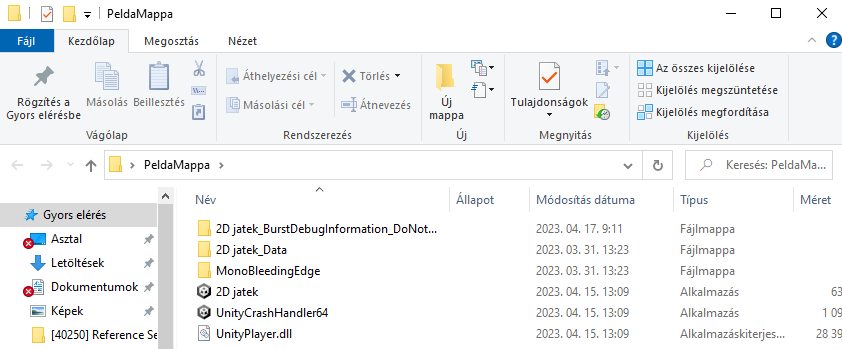
**Egy RAR fájlban található a program azt kell kicsomagolni egy csomagoló programmal pl.: WINRAR.**



**Ezt az opciót kell választanunk majd a kívánt cél mappába kicsomagolni.**



**Ezt követően kezdjük el a fájlok kicsomagolását:**

**Kicsomagolás után látható fájlok**:

Az exe futtatásával elindíthatjuk a játékot, további telepítést vagy beállítást nem igényel.

## Alkalmazás használata

Az exe elindítása után a főmenüben találjuk magunkat, ahol az options menüben részletesen le van írva az irányítás.

WASD vagy nyilak – ezekkel a gombokkal tud a karakterünk mozogni

Bal klikk – ennek a gomb lenyomásával tud a karakterünk támadni

Space-ennek a gomb lenyomásával tud a karakterünk ugrani

Play gomb rákattintása után elindul a játék

Az Options gomb rákattintása után az opciók vezérel a menü

**Az Options menü:**



Az Optionsben a Back gomb rákattintása után visszavisz minket a főmenübe

A Quit gomb rákattintása után bezárul a játék

**Escape lenyomása játék közben:**

A játék megáll, amíg a játékos újra le nem nyomja, vagy rá nem kattint az egyik opcióra.

A Resume gomb rákattintása után újra elindul a játék

A Menu gomb rákattintása után vissza visz minket a főmenübe

A Quit gomb rákattintása után bezárul a játék

## Játék ismertetése

Ez egy 2D-s platformer játék ahol a jelenlegi cél az hogy túljuss a pályán legyőzve az ellenfeleket és megölve a boss-t. Ezt a játékot bármilyen korosztály használhatja egyelőre kicsit kevesebb kihívást jelentő játékélményt biztosít, viszont a későbbiekben ezen javítani szeretnénk. Bárhol használható ez a játék videókártya nélkül is futtatható. Szórakozásra és kikapcsolódásra vágyoknak tökéletes unaloműző.

**Játék menete:**

Elindítjuk a játék programot.

Betöltés után megtekinthetjük a beállításokat, valamint elkezdhetjük a játékot.

A játék elindítása után egyetlen célunk, hogy végig vigyük a pályát legyőzve az ellenfeleket és végén a boss-t.

Miután legyőztük feldobja a game over menüt Victory felirattal, ahol kiválaszthatjuk, hogy kilépünk a játékból vagy vissza lépünk a főmenübe.

Főmenü kinézete, amikor belépsz a játékba (innen indul a játék):

Minden pályának más kinézete van a játékban és más ellenségek is találhatók rajta példa az első pályára:



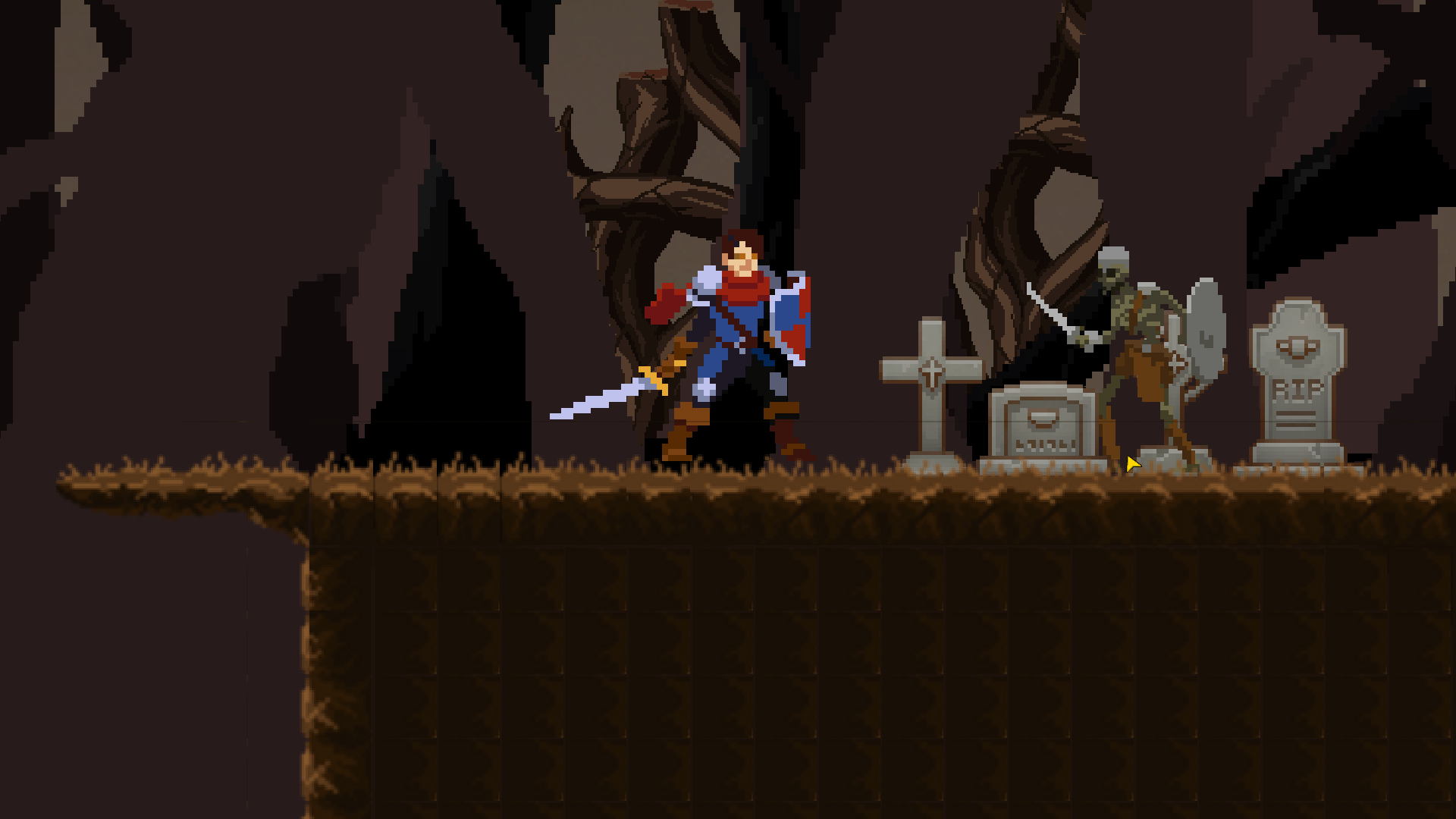
Elsőnek egy slimeal fog a karakterünk találkozni és őt kell legyőznie ő még nem támad vissza csak kezdőszörnynek van akit egyszerűen le kell győzni.

Ahogy tovább haladunk a pályával megjelenik más szörny is például:



Ez a szörny már képes érzékelni a karakter és meg is támadja viszont ő is könnyebben legyőzhető.

Második pályán egy csontváz szörny található, ami erősebb, mint az első pályákon lévő szörnyek ő is visszatud támadni illetve több életeröje és sebzése van. Példa:



Az utolsó pálya előtt egy felvezető pálya is van ahol nincs semmilyen ellenség vagy végig kell rajta menni:



A legvégső pálya ahol a boss is megtalálható így néz ki:



A boss máshogy érzékeli a karakterünket és egyből megtámadja ellenben az előző pályákon megtalálhatókkal. Ha a karakterünk meghal bármilyen küzdelemben a legutolsó checkpointon éled újra egyből neki nincs saját halál animációja.

A boss legyőzése után pedig megkapjuk a Game Over menüt

Escape lenyomásával előhozott a Pause menü kinézete:

# **Irodalomjegyzék**

## Internetes forrásaink

Akitől az animációkat és a mozgó platformot néztük:

<https://www.youtube.com/@codinginflow>

Akitől a támadást, bosst, menüvezérlést néztük:

<https://www.youtube.com/@Brackeys>

Ahonnan az asseteket néztük:

<https://assetstore.unity.com/>

Ahonnan az egyszerű hátterek berakását:

<https://www.youtube.com/watch?v=yuPpQoBJvvk>

Oldalak, amiket még használtunk:

<https://stackoverflow.com/>

<https://learn.unity.com/>

<https://hu.wikipedia.org/wiki/Kezd%C5%91lap>

## Könyvek

**Jamie Chan:** Learn C# in One Day and Learn It Well: C# for Beginners with Hands-On Project

**Andrew Stellman:** Head First C#

**Casey Hardman**: Game Programming with Unity and C#: A Complete Beginner’s Guide

**Jared Halpern**:Developing 2D Games with Unity: Independent Game Programming with C#

**Robert C. Martin:** Tiszta kód

**Franz Lanzinger:**2D Game Development with Unity

**Steve McConnell:** Code Complete

# **Mellékletek**

<https://github.com/Milaaan20/knights_quest.git>

Erről a linkről kell letölteni a játékhoz szükséges fájlokat. Letöltés után a fájlokat ki kell kicsomagolni a választott mappába.

# Összefoglalás

A projekt készítése során rengeteg tapasztalatot szereztünk arról, hogy milyen elkészíteni egy saját játékot és milyen nehézségekkel jár. A verzió, amit nyújtunk, az egy béta verziónak felel meg. Miután elkészítettük ezt a játékot elkezdtük jobban értékelni a játékokat a minket ért nehézségek révén. Sok élménnyel gazdagodtunk és új ismeretek szereztünk amelyeket következő játékkészítése során is alkalmazhatunk, mivel 2D-s játékot terveztünk ezért a megvalósítása sokkal több problémát jelentett számunkra mert nem volt annyi oktató videó és a Unity is a 3D-s fejlesztést támogatja.

1. Unity (forrás): <https://hu.wikipedia.org/wiki/Unity_(j%C3%A1t%C3%A9kmotor)> [↑](#footnote-ref-2)
2. Microsoft Visual Studio (forrás): <https://hu.wikipedia.org/wiki/Microsoft_Visual_Studio> [↑](#footnote-ref-3)
3. C# (forrás): <https://hu.wikipedia.org/wiki/C_Sharp> [↑](#footnote-ref-4)