**Debreceni SZC Beregszászi Pál Technikum és Kollégium**

**„Mystic Realm”**

**roguelike, platformer játék**

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  | Készítette:  Patai Szabolcs Lajos  Szilágyi Zsolt  Máté Patrik |

Debrecen

2024

Tartalomjegyzék

[1 Bevezetés 3](#_Toc164685698)

[2 Felhasználói dokumentáció 4](#_Toc164685699)

[2.1 Ismertető 4](#_Toc164685700)

[2.2 Rendszerkövetelmény 4](#_Toc164685701)

[2.3 Játék telepítése 5](#_Toc164685702)

[2.4 A játék 8](#_Toc164685703)

[2.4.1 Bejelentkezés & regisztráció 8](#_Toc164685704)

[2.4.2 Főmenü 8](#_Toc164685705)

[2.4.3 A játékmenet 8](#_Toc164685706)

[2.4.4 Irányítás 8](#_Toc164685707)

[2.4.5 Ellenfelek 9](#_Toc164685708)

[2.4.6 Szintlépés 9](#_Toc164685709)

[2.5 Mentett adatok 9](#_Toc164685710)

[3 Fejlesztői dokumentáció 9](#_Toc164685711)

[3.1 Alkalmazott fejlesztői eszközök 9](#_Toc164685712)

[3.1.1 Unity 9](#_Toc164685713)

[3.1.2 Fejlesztői környezet 9](#_Toc164685714)

[3.1.3 Inno Setup 11](#_Toc164685715)

[3.2 Adatmodell 12](#_Toc164685716)

[3.2.1 Users 12](#_Toc164685717)

[4 Weboldalt készítette: Patai Szabolcs Lajos 13](#_Toc164685718)

[5 Knightot, menüt, tutorialt és az arena2t készítette: Szilágyi Zsolt 13](#_Toc164685719)

[5.1 Használt programok 13](#_Toc164685720)

[5.2 Bejelentkezés és regisztráció 13](#_Toc164685721)

[5.3 Menü 14](#_Toc164685722)

[5.4 Karakter választás 15](#_Toc164685723)

[5.5 Knight 16](#_Toc164685724)

[5.6 Tutorial 17](#_Toc164685725)

[5.7 Arena2 18](#_Toc164685726)

[6 Archert, arena1et, arena3at és az UnderTaleEventet készítette: Máté Patrik 19](#_Toc164685727)

[6.1 Használt programok 19](#_Toc164685728)

[6.2 Archer 19](#_Toc164685729)

[6.3 Arena1 22](#_Toc164685730)

[6.4 Arena3 24](#_Toc164685731)

[6.5 UnderTaleEvent 25](#_Toc164685732)

[7 Tesztelés és eredmények 26](#_Toc164685733)

[8 Összegzés 28](#_Toc164685734)

[9 Forrásjegyzék 29](#_Toc164685735)

# Bevezetés

A záróvizsgánk beadandójára egy 2D-s roguelike, platformer játékot és hozzá egy weboldalt készítettünk, ahol le lehet tölteni a játékot, és további információ található meg a játékról. A játék Unity fejlesztőkörnyezetben, illetve C# programozási nyelvben valósítottuk meg. A weboldal PHP, illetve CSS programozási nyelveken íródott. A játék célja, hogy éld túl a meghatározott ideig. A játék folyamán szörnyek megölésével pénzt szerezhetsz, amivel fejlesztheted a karaktered. A választásunk azért esett erre a projektre mivel korábban is érdekelt minket a játékfejlesztés és szerettünk volna mi is megtudni, belelátni hogyan is készítenek ilyen fajta játékokat.

A Roguelike (vagy rogue-like ) a [szerepjáték](https://en.wikipedia.org/wiki/Role-playing_game) egy stílusa, amelyet hagyományosan a [pályákon](https://en.wikipedia.org/wiki/Level_(video_gaming))[való átjárás](https://en.wikipedia.org/wiki/Dungeon_crawl), [körökre osztott játékmenet](https://en.wikipedia.org/wiki/Turns,_rounds_and_time-keeping_systems_in_games), rács alapú mozgás és a [játékos karakterének](https://en.wikipedia.org/wiki/Player_character)[végleges halála](https://en.wikipedia.org/wiki/Permanent_death) jellemez . A legtöbb roguelike egy [nagy fantáziaalapú](https://en.wikipedia.org/wiki/High_fantasy) narratíván alapul[[1]](#footnote-1).

A platformjáték (más néven platformer vagy Jump ’n’ Run) egy videójáték-műfaj, az akciójáték alműfaja. A játékos által irányított karakternek platformokon keresztül kell ugrálnia és/vagy különböző akadályokat kell átugrania. A játékos feladata, hogy a karakterével megfelelő időben ugorjon, hogy tovább tudjon menni vagy ne essen le. Az ugráson kívül más mozgáselem is szerepelhet, mint az úszás, mászás vagy repülés. Platformjátéknak nevezhető az olyan játék, aminek szerves részét képezi a platformokon való ugrálás[[2]](#footnote-2).

Az oldalnézetes vagy 2D-s játékok legfőbb tulajdonsága a kétdimenziós grafika. Az FPS és a TPS játékok előtt ez volt a legelterjedtebb és legváltozatosabb műfaj. Az oldalnézetes játékok gyakorlatilag minden platformon megjelentek és rengeteg típusuk van. Gyakran készültek ilyen játékok sikeres filmek és rajzfilmek alapján. A kétdimenziós grafika egyértelmű előnye, hogy nem kell nézőpontokkal, kameramozgatással foglalkozni. A játékos legtöbbször oldalnézetben látja a pályát, mozgás közben pedig a pálya együtt halad a játékfigurával (side-scrolling). A látható világot kézzel vagy számítógéppel rajzolták[[3]](#footnote-3).

# Felhasználói dokumentáció

## Ismertető

A játék egy 2D-s Roguelike játék. Csak egyjátékos módban lehet játszani. A játékban két karakter közül lehet választani, amiket irányíthatunk, több különböző ellenféllel találkozhatunk, amiket meg kell, hogy öljünk. 3 különböző pályán kell keresztülmennie a játékosnak, amelyek fokozatosan nehezednek.

Az ellenfelek megölésével pénzt kapunk. A pénz segítségével a boltban fejleszthetjük a karakterünk életerejét, sebzését, kritikus találati esélyt és annak sebzését. Az első főellenség a játékban egy óriási slime, ami lézereket küld felénk, amiket ki kell kerülnünk ameddig a zene véget nem ér.

Ha meghalunk vagy leesünk előröl kell kezdenünk a megkezdett pályát. A játékosnak egy élete van, amit elveszíthet a játék során. A karakterünk az ellenségek ütközésekor sebzést szenved el. Ha a játékos kiesik a pályáról meghal.

A pálya teljesítéséhez túl kell élni a meghatározott ideig, amit a képernyő tetején láthatunk. Miután lejárt az idő, megjelenik egy ajtó, átdob minket egy a pályához tartozó platformer részhez. Ezt a platformer pályarészt úgy teljesíthetjük, hogy végig megyünk a pályán. Ha ez teljesült akkor feloldjuk a következő pályarészt.

## Rendszerkövetelmény

A játék asztali számítógépen futtatható. A részletes rendszerkövetelmények megtalálhatók a Unity honlapján[[4]](#footnote-4). A játék futtatásához a következőek a minimum követelmények:

• Operációs rendszer: Windows 7(SP1+) vagy újabb

• Processzor: 2Ghz, ami támogatja a SSE2 technológiát

• Memória: 512MB

• Videókártya: DirectX10 vagy jobb

• Tárhely: 100MB szabad hely

• Internet elérés

Ajánlott rendszerkövetelmény:

• Operációs rendszer: Windows 10 vagy újabb

• Processzor: Intel Core i3-12100F 3.30GHz

• Memória: 2GB

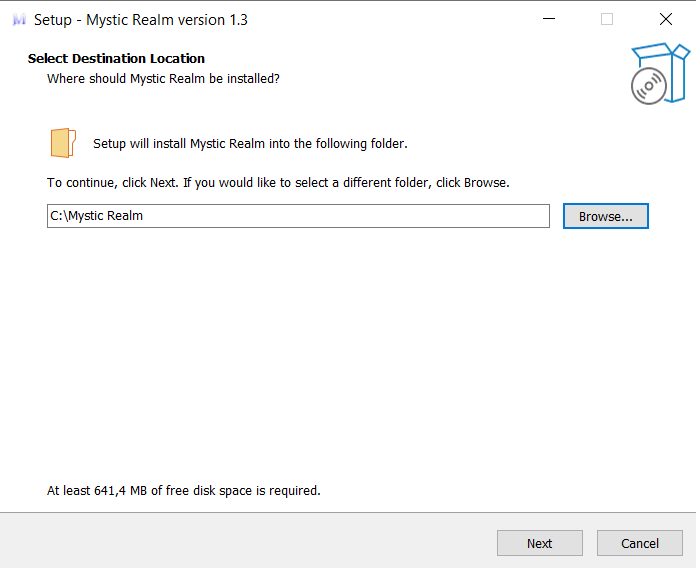
• Videókártya: GIGABYTE GeForce GT 1030 2GB GDDR5

• Tárhely: 100MB szabad tárhely

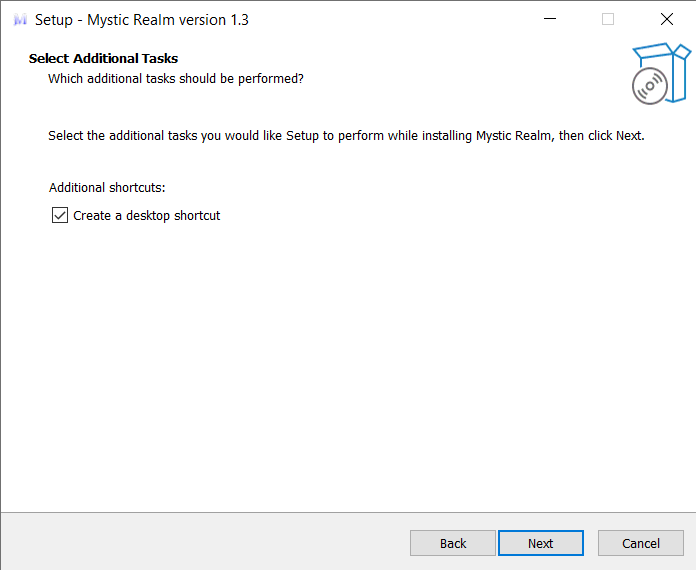
• Internet elérés

## Játék telepítése

Indítsd el a „Mystic Realm Setup.exe”-t, majd válaszd ki hova szeretnéd telepíteni a játékot.

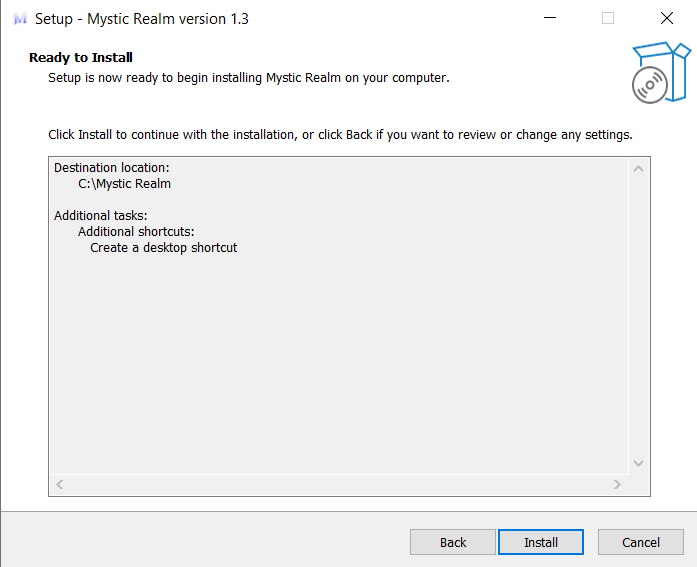


1. *ábra – Telepítési folyamat 1*

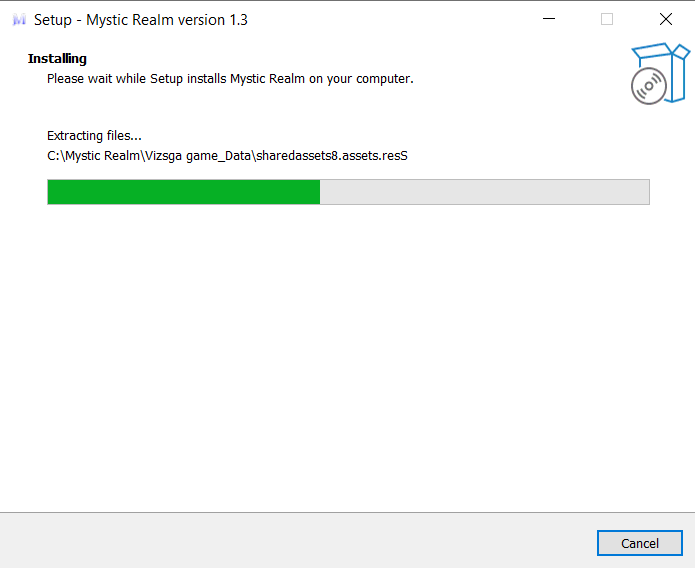
A telepítő felajánlja, hogy készítsen gyorselérés ikont az asztalra. Ez tetszőlegesen elfogadható.

1. *ábra – Telepítési folyamat 2*

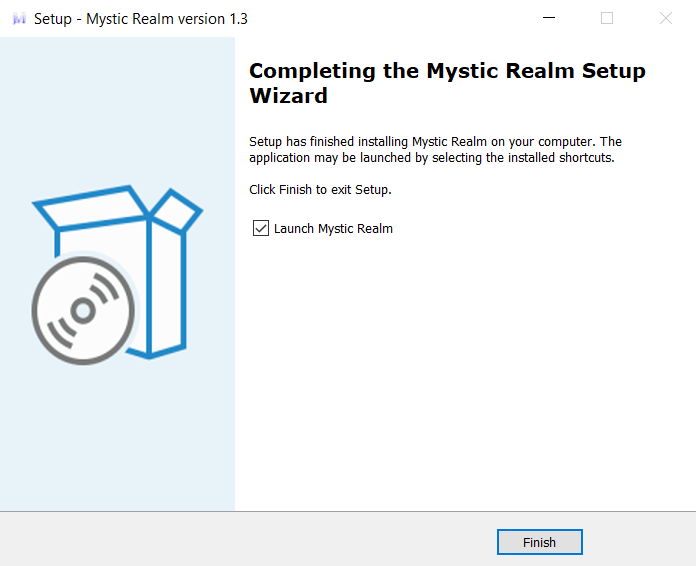
A telepítő készen áll a telepítésre. Nyomd meg az [Install] gombot.



1. *ábra – Telepítési folyamat 3*

A telepítés folyamatban van, várd, meg míg befejeződik.

1. *ábra – Telepítési folyamat 4*

A telepítés befejeződött, a telepítő felajánlja, hogy a [Finish] gomb lenyomása után elindítsa-e a játékot. Ez tetszőlegesen elfogadhat. Nyomj a [Finish] gombra. Kész vagy.

1. *ábra – Telepítési folyamat 5*

## A játék

Indítsa el az asztalon található Mystic Realm parancsikont, vagy ha nem jelölte be a telepítésnél, hogy készítsen gyors elérésű ikont akkor a telepítés helyén lévő Mystic Realm mappában található Mystic Realm.exe-t indítsa el.

### Bejelentkezés & regisztráció

A játék elindítása után a bejelentkezés panelt fogod látni. A panelen kettő lehetőség adott, be tudsz jelentkezni a meglévő fiókodba vagy tudsz egy új fiókot létrehozni. Ha nincs, fiókod akkor Kattints a [Register] gombra.

### Főmenü

Bejelentkezés után a főmenüt látod. Itt a játék négy fő opciói közül választhatsz. Itt az első a [New Game] gomb, amellyel elkezdődhet a játék. Következő gomb a [Load Game] ahol folytathatod az eddig mentett játékodat. A következő gomb az [Settings] ahol be tudod állítani a neked megfelelő beállításokat. A negyedik gomb a [Exit] amivel be tudod zárni a játékot.

### A játékmenet

Miután megnyomtad a [New Game] gombot, elindul a játék és a kiképző pályán találod magad, ahol megtanulhatod a játék alapvető funkcióit. A cél, hogy túl éld a meghatározott ideig. A játék teljesítése során ellenfelekkel és szakadékokkal találkozhatsz, amik megnehezítik a pálya teljesítését.

### Irányítás

A játékban a karaktert kettő féle irányítást tesz lehetővé.

Nyilak segítségével:

* ←: a balra nyíl lenyomásával balra megy a karakter
* →: a jobbra nyíl lenyomásával jobbra megy a karakter
* ↑: felfele nyíl lenyomásával ugrik egyet a karakter

A, D, W, SPACE segítségével:

* A: az A gomb lenyomásával balra megy a karakter
* D: a D gomb lenyomásával jobbra megy a karakter
* W: a W gomb lenyomásával ugrik egyet a karakter
* SPACE: a SPACE gomb lenyomásával aktiválod a speciális képességét a karakternek

### Ellenfelek

Az első pályán slime, a másodikon skeleton, a harmadikon pedig további 4 ellenség és 2 főellenség található. Az ellenfelek karaktert követik. Hozzájuk érve a karakterünk sebződik. A pályákon az ellenfelek, minden egyes perccel egyre erősebbé válnak.

### Szintlépés

Az ellenfelek megölésével halálukkor tapasztalati pontot adnak, ha összegyűjtöttél elegendő tapasztalati pontot szintet lépsz, illetve a 3 pályán te választhatod ki, hogy melyik statisztikádat erősíted.

## Mentett adatok

A pénzed akkor kerül mentésre, ha a karaktered meghal vagy kilépsz a játékból. A bolt bezárásakor mentésre kerül az adatok. A játék újraindításával nem vesződik el a pénzed. Az adataidat úgy tudod törölni, hogy a menüben rányomsz a [New Game] gombra.

# Fejlesztői dokumentáció

## Alkalmazott fejlesztői eszközök

### Unity

A Unity egy videójáték-motor, amelyet a Unity Technologies fejleszt. A Unity segítségével háromdimenziós, illetve kétdimenziós videójátékokat, ezen kívül egyéb interaktív jellegű tartalmakat lehet létrehozni. 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, iPhone vagy Android alatt. A Unity-nek 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 segít.[[5]](#footnote-5)

### Fejlesztői környezet

Az integrált fejlesztői környezet, vagyis rövidítve IDE (Integrated Development Environment) egy olyan egy olyan szoftver, amelyben a megírni kívánt forráskódot lehet szerkeszteni. Az IDE nem csak a forráskód szerkesztési lehetőségét adja, hanem számos beépített funkciót tartalmaz. Főbb eszközei, funkciói:

* Szintaxis kiemelése: Az IDE-szerkesztő általában szintaktikai kiemelést biztosít, ilyenkor mind a vizuálisan különböző színekkel és betű-hatásokkal rendelkező szintaktikai hibákat képes megjeleníteni.
* Kód kiegészítés: a kódkiegészítés egy fontos IDE funkció, amelynek célja a programozás felgyorsítása. A modern IDE-knek már van intelligens kód befejezése is. Az intelligens kódkiegészítése egy környezet tudatos funkció, amely felgyorsítja az alkalmazások kódolási folyamatát az elírások és más gyakori hibák csökkentésével. Erre általában gépelés közbeni automatikus kiegészítés felugró ablakon keresztül történik.
* Hibakeresés: Az IDEs-t hibakeresésre is használják, az integrált hibakeresőt használva, támogatva a töréspontok beállítását, amely a kód adott részénél megállítja a programot így megtudhatjuk hol jelenetezik a hiba vagy mi okozza azt, illetve nyomon követhetjük a változók értékeit is.
* Vizuális programozás: A Visual Basic lehetővé teszi a felhasználók számára, hogy új alkalmazásokat hozzanak létre programozás, építőelemek vagy kód csomópontok áthelyezésével, hogy folyamatábrákat vagy szerkezet diagramokat hozzon létre, amelyeket aztán lefordítanak vagy értelmeznek.

A Visual Studio a Microsoft integrált fejlesztői környezete (IDE). Számítógépes programok, valamint webhelyek, webes alkalmazások, mobil alkalmazások fejlesztésére szolgál. A Visual Studio 36 különböző programozási nyelvet támogat, és lehetővé teszi, hogy a kódszerkesztő és a hibakereső támogassa szinte bármilyen programozási nyelvet, feltéve, hogy létezik nyelvspecifikus szolgáltatás.

A Visual Studio-nak a legnagyobb változást a .NET-keretrendszert használó felügyelt kódfejlesztő környezet bevezetése jelentette. A .NET használatával fejlesztett programokat nem gépi nyelvre fordítják (mint például a C++), hanem a Microsoft Intermediate Language (MSIL) vagy a Common Intermediate Language (CIL) formátumra. Amikor egy CILalkalmazás fut, akkor azlefordítása közben a végrehajtás alatt álló platformnak megfelelő gépi nyelvre kerül, ezáltal a kód több platformon is hordozhatóvá válik. A CIL-be fordított programok csak olyan platformokon futtathatók, amelyek rendelkeznek Common Language Infrastructure implementációval.

A .NET keretrendszer azon részét, amely elvégzi a konverziót köztes kód és a gépikód között, közös futtató környezetnek Common Language Runtime (CLR) nevezzük. Ennek a megoldásnak az igazi ereje abban rejlik, hogy a .NET rendszerben támogatott összes nyelv először MSIL-re fordul. Így a CLR számára teljesen mindegy, hogy a kód, amelyet lefordít, eredetileg Visual Basic .NET, J#, vagy esetleg C# nyelven volt megírva. Az IL tehát elfedi a .NET által támogatott nyelvek közötti különbségeket is.[[6]](#footnote-6)

A C# a Visual Basic mellett a .NET fő programozási nyelve. 1999 –benn Anders Hejlsberg vezetésével kezdték meg a fejlesztését. A C# tisztán objektumorientált, típusbiztos, általános felhasználású nyelv. A tervezésénél a lehető legnagyobb produktivitás elérését tartották szem előtt. A nyelv elméletileg platform független (létezik Linux és Mac fordító is), de napjainkban a legnagyobb hatékonyságot a Microsoft implementációja biztosítja.

Így a választásunk a Visual Studio-ra esett mivel a „Development with Unity” kiegészítőt tartalmazza, amely javítja a Unity C# szkriptek írását és hibakeresését, valamint a Unity projektekkel való munkát.

* A Unity projektekhez hangolt hibakereső segítségével könnyen elhárítható a hibás kód.
* Fedezze fel írjon gyorsan Unity szkripteket a Unity-specifikus IntelliSense kódkiegészítés segítségével.
* Tudjon meg többet az ön által írt kódról a Unity dokumentációjának gyors elérésével.
* Írjon jobb kódot az újrafaktorálási beállításokkal, amelyek követik a Unity szkriptekre vonatkozó bevált módszereket.
* Határozza meg, hogyan hívja meg a Unity-motor a kódot a CodeLens tippekkel az üzenetfunkciókra és az eszközhasználatra vonatkozóan.

### Inno Setup

Az installer (telepítő) elkészítéséhez az Inno Setup[[7]](#footnote-7) programot használtam, amellyel egyszerűen és gyorsan létre tudtam hozni a telepítőt. A program az indító exe és a hozzátartozó kiegészítő fájlok összetömörítés segítségével készíti el a telepítőt. A programban lehetőségünk van megadni az alkalmazás nevét, verzió számát, kiadóját, és weblapját, illetve, hogy milyen nyelven hozza létre. Még meg tudjuk adni, hogy készítsen-e gyors elérésű ikont az asztalra. És még számos funkció elérhető számunkra.

* + 1. Assetek

Még megemlíteném a felhasznált asseteket a játékunkban. Mivel nem állt rendelkezésre technikai eszköz, illetve grafikai tudásunk sem a legjobb így úgy döntöttünk, hogy a Unity Asset Store-ból illetve az itch.io-ról választjuk ki az ingyenes felhasználható asseteket. Az egyik ilyen a karakter asset a Free Game Assets (GUI, Sprite, Tilesets) által készített „Free Fantasy Knight”[[8]](#footnote-8). A másik karaktert pedig a Kronovi- által készített „Archer Hero A mysterious Archer animated character”[[9]](#footnote-9) -ből választottuk.

## Adatmodell

|  |  |
| --- | --- |
| **Oszlop** | **Típus** |
| userid (elsődleges) | int(10) |
| username | varchar(200) |
| password | varchar(150) |
| email | varchar(100) |

1. *ábra – Adatbázis tábla*

### Users

A users táblában a felhasználók adatait tároljuk.

* A tábla első mezője az userid ez egy szám típusú mező. Ez a tábla elsődleges kulcsa, ez automatikus generálódik mikor egy új felhasználó regisztrál. Ez a mező azért fontos mivel az adatok lekérdezése során ezzel tudjuk beazonosítani a felhasználót.
* A második mező a username ami egy varchar típusú mező ami szöveget és számot is tartalmazhat. Ez egy másodlagos megkülönböztetés a felhasználók között.
* A harmadik mező a password ami szintén egy varchar típusú mező. Ebben a mezőben a jelszó titkosított formában kerül be.
* A negyedik mezőbe az email cím található, amely varchar típúsban van. Itt a felhasználók email címe van eltárolva

# Weboldalt készítette: Patai Szabolcs Lajos

# Knightot, menüt, tutorialt és az arena2t készítette: Szilágyi Zsolt

## Használt programok

A felhasznált szoftvereket, amit használtam az asztali alkalmazás elkészítéséhez az a Unity és a Visual Studio 2022. Azért a használom a Unity-t, mivel népszerű, egyszerű a használata és C# programozási nyelvet igényel. Valamint a 2D-s játékokhoz egyik, ha nem a legjobb játékmotor. A Visual Studio 2022-t azért használom mert tartalmazza a „Development with unity” kiegészítőt, amely javítja a Unity C# szkriptek írását és hibakeresését, valamint a Unity projektekkel való munkát.

## Bejelentkezés és regisztráció

Kezdeném is a bejelentkezés és regisztrációhoz köthető szkriptekkel. A Registration szkript tartalmazza a weboldalunkhoz való eljutást, ahol lehetőség van regisztrálni. A Login osztály, amely felelős a bejelentkezésért.

A Login szkriptben találhatóak a bejelentkezéshez tartozó InputField-ek, gombok. Mikor a felhasználó beírja az adatait akkor az InputField-ek eltárolják azokat, a Login, illetve Register gomb lenyomásakor a UnityEvent.AddListener használatával kiterjeszti az OnClick kattintási eseményt. A hozzáadott UnityAction a gombnyomás felengedésekor kerül meghívásra a benne található rész. A StartCoroutine használatakor a Unity egy új Coroutine típusú objektumot hoz létre, ez az objektum végrehajtja a benne lévő műveletet, amely a Main szkript Instance változó Web komponens, illetve Web szkriptben lévő LoginPlayer IEnumerator interfésznek adja át az InputField-ben lévő adatokat.

A Web szkriptben a bejelentkezést a UnityWebRequest kezeli a http kommunikáció folyamatát a webszerverekkel. Az adatok letöltéséhez és feltöltéséhez használja a DownloadHandler és az UploadHandler alkalmazást. A UnityWebRequest statikus segédprogramokat tartalmaz, amelyek a gyakori használati esetekre konfigurált UnityWebRequest példányokat adják vissza.

A [Register] gomb OnClick eseményén meghívjuk az OpenWeb() metódust a Registration szkriptből. Aminek következtében megnyílik a weboldalunk.

1. A képen szöveg, képernyőkép, Betűtípus, Márka látható

   Automatikusan generált leírás*ábra – Bejelentkezés és regisztráció*

## Menü

A menüben a SceneLoder szkriptet használjuk, amiben 7 metódus található. Ehhez a szkripthez szükségünk van a SceneManagement névtérből származó SceneManager osztályra, hogy jelenetet tudjunk váltani. A [New Game] gomb OnClick eseményén meg van hívva a NewGame() metódus, amely kitörli az összes mentett adatot és jelenetet vált.

A [Load Game] gomb OnClick eseményén a LoadGame() metódus van meghívva, amely megvizsgálja, hogy található-e mentett adat, és ha van akkor betölti azt.

A [Settings] gomb OnClick eseményén a Settings() metódus lesz meghívva amely előhozza a beállításokat.

A [Exit] gomb OnClick eseményén a QuitGame() metódus van meghívva amelyben az Application.Quit() parancsot használjuk.

1. A képen képernyőkép, fa, Digitális képszerkesztés, Számítógépes játék látható

   Automatikusan generált leírás *ábra – Menü*

## Karakter választás

A képen szöveg, rajzfilm, Grafikus tervezés, képernyőkép látható

Automatikusan generált leírás Itt a CharacterSelect szkriptet használjuk, amiben 2 metódus található. A KnightSelect() és az ArcherSelect() amiben elmentjük a választott karaktert, és átvisz a Tutorial jelenetbe.

1. *ábra – Karakter választás*

## Knight

Ennek a karakternek az elkészítéséhez 5 szkriptet használtam. Ezek a KnightAttack, KnightDeath, KnightHealth, KnightMovement és a KnightXP.

A képen szöveg, képernyőkép, szoftver, Multimédiás szoftver látható

Automatikusan generált leírás

1. *ábra – Kódrészlet a KnightDeathből*

A KnightDeath szkriptben globális változókat használok:

* float timer - idő
* KnightMovement knightMovement – meghívom a KnightMovement szkriptet
* GameObject deadpanel – megadom a deadpanel nevű gameobjectet
* GameObject HUD – megadom a HUD nevű gameobjectet
* GameObject music – megadom a music nevű gameobjectet
* KnightHealth knightHealth - meghívom a KnightHealth szkriptet
* AudioManager audioManager - meghívom a AudioManager szkriptet

A Start() metódusban a music gameobjectet eltüntetem, és elindítom a halál hangot. Az Update() metódusban vizsgálom hogy a karakter életereje kisebb vagy egyenlő mint 0. Ha kisebb vagy egyenlő akkor elindítok egy időzítőt, és lelassítom az időt. Elindítom a „Death” animációt, és az alive bool változót false-ra állítom. Amikor a timer nagyobb lesz mint 1 megjelenítem a deadpanelt, eltüntetem a HUD-ot és visszaállítom az idő sebességét.

## A képen szöveg, képernyőkép, szoftver látható Automatikusan generált leírásTutorial

1. *ábra – Kódrészlet a LevelSelectorDoor-ból*

A LevelSelectorDoor szkriptben globális változókat használok:

* Animator anim – megadom az Animatort
* SaveManager saveManager - meghívom a SaveManager szkriptet

A Start() metódusban meghívom az Animatort. Az OnTriggerStay2D() metódusban megvizsgálom, hogy a collison tagje az a „Player”-e. Ha igen, az „Open” animációt elindítom. Ezen kívül ha a játékos megnyomja az „E” betűt akkor meghívom a savemanager TutorialSave() metódusát, és betöltöm a „Level Selector” jelenetet.

Az OnTirggerExit2D() metódusban megvizsgálom, hogy a collison tagje az a „Player”-e. Ha igen, az „Open” animációt leállítom.

## A képen szöveg, képernyőkép, szoftver látható Automatikusan generált leírásArena2

1. *ábra – Kódrészlet a CountDown-ból*

A CountDown szkriptben globális változókat használok:

* GameObject door - meghívom a door nevű gameobjectet
* float countDown - visszaszámláló
* float min - perc
* Text minutes - megadom a minutes Text-et
* Text seconds - megadom a seconds Text-et

Az Update() metódusban létrehozok egy privát double „b” nevű változót és egy Math.Round() metódus segítségével egészre kerekítem a coundown értékét. A min változó Mathf.FloorToInt() metódust használok, hogy megkapjam a percet. Ezután létrehozok egy privát float „sec” változót, aminek megadom értéknek a Mathf.FloorToInt() segítségével a másodpercet. A percet stringé változtatom és megadom a minutes értékének, majd ezt ugyanúgy megcsinálom a másodpercel.

Vizsgálom, hogy a countdown értéke nagyobb vagy egyenlő mint 0.5. Ha igen akkor a coundown értékét folyamatosan csökkentem. Ha a feltétel hamis akkor a door-t megjelenítem és ezt a gameobjectet eltüntetem.

# Archert, arena1et, arena3at és az undertale eventet készítette: Máté Patrik

## Használt programok

A felhasznált szoftvereket, amit használtam az asztali alkalmazás elkészítéséhez az a Unity és a Visual Studio 2022. Azért a használom a Unity-t, mivel népszerű, egyszerű a használata és C# programozási nyelvet igényel. Valamint a 2D-s játékokhoz egyik, ha nem a legjobb játékmotor. A Visual Studio 2022-t azért használom mert tartalmazza a „Development with unity” kiegészítőt, amely javítja a Unity C# szkriptek írását és hibakeresését, valamint a Unity projektekkel való munkát.

## Archer

Ennek a karakternek az elkészítéséhez 7 szkriptet használtam. Ezek a ArcherAttack, ArcherDeath, ArcherHealth, ArcherMovement, ArcherXP, ArrowSpawner és az ArrowMovement.

A képen szöveg, képernyőkép, szoftver látható

Automatikusan generált leírás

1. *ábra – Kódrészlet a ArcherAttack-ból*

Az ArcherAttack szkriptben globális változókat használok:

* bool skillActive – jelzi, hogy a skill aktív-e
* float damage – sebzés értéke
* float timer - időzítő
* float timer2 – 2. időzítő
* float attackSpeed – támadási sebesség
* float critRate – kritikus támadási esély
* float critDMG – kritikus sebzés
* float animwait – animáció időzítője
* float mana – mana értéke
* float maxMana – maximum mana értéke
* float manaCost – archer képességének manaköltsége
* Slider Mana – mana csúszkája
* ArcherHealth archerHealth – meghívom az ArcherHealth szkriptet
* ArcherMovement archerMovement - meghívom az ArcherMovement szkriptet
* HealthExecutioner healthExecutioner - meghívom az HealthExecutioner szkriptet
* ArrowSpawner arrowSpawner - meghívom az ArrowSpawner szkriptet
* AudioManager audioManager - meghívom az AudioManager szkriptet

A Start() metódusban inicializálom az archerHealth-et és az archerMovement-et. Az Update() metódusban elindítom a timer2-t és az attackSpeed timert. A Mana csúszka maximum értékének megadom a maximum mana értékét, és az aktuális manának megadom a mana értékét, meghívom a ManaUsage() metódust. Ha az attackSpeed nagyobb vagy egyenlő mint 0.5 másodperc meghívom az Attack() metódust. Egy if-ben megnézem, hogy a skillActive bool változó igaz-e, mana nagyobb mint 0 és a karakter életereje nagyobb mint 0, akkor az aktuális manából folyamatosan kivonom a manaCost-ot. Ha ez az if nem igaz ekkor a skillActive boolt false-ra állítom.

A ManaUsage() metódusban egy if segítségével megvizsgálom hogy a játék megnyomja-e a [Space] billentyűt amivel a skillActive boolt true-ra állítja. Else if segítségével megnézem, hogy a skillActive bool igaz-e, ha igaz és a játék megnyomja a [Space] billentyűt akkor a skillActive boolt false-ra állítja.

Az Attack() metódusban if segítségével megvizsgálom hogy a játékos megnyomta-e a bal egérgombot és a „Run” animáció nem megy, akkor elindítom az animáció időzítőjét és az „Attack” animációt true-ra állítom. Ezután még egy if-el megnézem, hogy az animáció időzítője nagyobb mint 0.75 másodperc, ha igen akkor lejátszom a támadás hangeffektjét és meghívom a SpawnArrow() metódust amivel létrehozok egy nyilat. Egy if-ben megnézem, hogy a skillActive változó igaz-e, akkor az animáció időzítőjét 0.5 másodpercre állítom, ha nem igaz akkor 0-ra. Egy else if használatával megvizsgálom, hogy a játékos felengedte-e a bal egérgombot. Mikor felengedte akkor az „Attack” animációt false-ra, a támadási sebességet 0-ra és az animáció időzítőjét is 0-ra állítom.

## A képen szöveg, képernyőkép, szoftver látható Automatikusan generált leírásArena1

1. *ábra – Kódrészlet a SlimeXP-ből*

Az SlimeXP szkriptben globális változókat használok:

* CountDown countDown - meghívom az CountDown szkriptet
* ClassLoader classLoader - meghívom az ClassLoader szkriptet
* KnightXP knightXP - meghívom az KnightXP szkriptet
* ArcherXP archerXP - meghívom az ArcherXP szkriptet
* SlimeHealth slimeHealth - meghívom az SlimeHealth szkriptet
* SlimeDamage slimeDamage - meghívom az SlimeDamage szkriptet
* int slimeXP – slime XP adása
* int slimeCoin – slime pénz adása
* int slimeLevel – slime szintje
* float timer - időzítő
* int minutes - percek

A Start() metódusban megkeresem a „LogicManager”, „CountDown” és a „Player” gameobjectet és inicializálom a classLoader-t, slimeHealth-et, slimeDamage-et, countDown-t. Egy if használatával megvizsgálom hogy a játékos melyik karaktert választotta és ez alapján inicializálom az adott karakter XP szkriptjét, és meghívom a SlimeLevelUp() metódust.

A SlimeLevelUP() metódusban while ciklusban megvizsgálom hogy a CountDown szkriptben lévő min változó értéke kisebb mint a minutes változó értéke. Ha kisebb akkor hozzáad a slime szintjéhez egyet és a slime pénzét megszorozza a slime szintjével. Majd egy switch segítségével a slime életerejét szintenként megnövelem, a sebzését pedig megszorzom a szintjével és a minutes változóból kivonok egyet.

A SlimeGiveXP() metódus segítségével amikor meghal a slime akkor az adott karakternek hozzáadom az XP-éhez a slimeXP változó értékét.

A SlimeGiveGold() metódus használatával amikor meghal a slime akkor az adott karakternek hozzáadom a pénzéhez a slimeCoin változó értékét, a karakter pénzét átalakítom string típussá és kiíratom.

## A képen szöveg, képernyőkép, szoftver látható Automatikusan generált leírásArena3

1. *ábra – Kódrészlet a Pistol-ból*

A Pistol szkriptben globális változókat használok:

* PlayerStats playerStats - meghívom az PlayerStats szkriptet
* BulletSpawner bulletSpawner - meghívom az BulletSpawner szkriptet
* GameObject crosshair - meghívom az GameObject szkriptet
* AudioManager audioManager - meghívom az AudioManager szkriptet
* float timer - időzítő

A Start() metódusban megkeresem a „Player” meg a „Crosshair” gameobjectet, és inicializálom a playerStats, bulletSpawnert.

Az Update() metódusban elindítok egy időzítőt. Egy if használatával megvizsgálom, hogy a játékos megnyomta-e a bal egérgombot és az időzítő nagyobb, mint a PlayerStats szkripben lévő fireRate változó értéke. Ha igaz akkor lejátszom a pisztoly lövésének hangeffektjét, meghívom a BulletSpawner szkriptben lévő Bullet() metódust amivel leidézek egy golyót és az időzítőt lenullázom. Egy másik if-el pedig megoldom, hogy a karakter a célkereszt irányába nézzen.

## A képen szöveg, képernyőkép, szoftver, Multimédiás szoftver látható Automatikusan generált leírásUnderTaleEvent

1. *ábra – Kódrészlet a SlimeKingDeath-ből*

A SlimeKingDeath szkriptben globális változókat használok:

* float timer -t indőzítő
* bool slimeKingAlive – jelzi, hogy a SlimeKing él-e
* SaveManager saveManager - meghívom az PlayerStats szkriptet
* GameObject text – megadom a text nevű gameobjectet
* UnderTaleHP playerHP - meghívom az PlayerStats szkriptet
* Animator anim – megadom a SlimeKing animátort

A Start() metódusban inicializálom az Animatort.

Az Update() metódusban ha a játékos életben van elindítok egy időzítőt. Ha az időzítő értéke nagyobb mint 118 másodperc akkor a „Death” animációt elindítom és a slimeKingAlive boolt false-ra állítom. Amikor az időzítő értéke nagyobb lesz mint 120 másodperc a SaveManager szkriptben lévő saveWorldBoss1() metódust meghívom és megjelenítem a text szövegét. Mikor az időzítő nagyobb lesz mint 123 másodperc akkor betöltöm a következő jelenetet.

# Tesztelés és eredmények

1. Bejelentkezés és regisztráció:

* Elvárt kimenet: Helyesen kitöltött beviteli mezőkkel a Login gomb lenyomásakor bejelentkezik
* Eredmény: A Login gomb lenyomásakor sikeresen bejelentkezett
* Elvárt kimenet: A regisztráció gomb lenyomáskor behozza a weboldalunkat
* Eredmény: A regisztráció gomb lenyomáskor sikeresen betöltötte a weboldalunkat
* Elvárt kimenet: Helytelen kitöltött beviteli mezőkkel a Login gomb lenyomásakor kiírja „Wrong username or password” szöveget
* Eredmény: Sikeresen kiírja az elvárt szöveget

1. Menü:

* Elvárt kimenet: New Game gombra kattintva új játékot kezd és kitöröl minden mentett adatot
* Eredmény: Sikeresen kitöröl minden mentett adatot és új játékot kezdett
* Elvárt kimenet: A Load Game gombra kattintva folytatja a megkezdett játékot
* Eredmény: Betöltötte a mentést
* Elvárt kimenet: A Settings gombra kattintva megjeleníti a beállítások menüt
* Eredmény: Sikeresen megjelenítette a beállításokat
* Elvárt kimenet: Exit gombra kattintva a játék bezáródik
* Eredmény: A játék bezáródott

1. Beállítások:

* Elvárt kimenet: Az Apply gomb lenyomásakor lementi a beállított értékeket
* Eredmény: Sikeresen lementette az értékeket
* Elvárt kimenet: A Default gomb lenyomásakor visszaállít minden beállítást az alapértékre
* Eredmény: Sikeresen visszaállt minden az alapértékre
* Elvárt kimenet: A Back gombra kattinva visszakerülünk a menübe
* Eredmény: Sikeresen visszakerültünk a menübe

1. Karakter választó:

* Elvárt kimenet: A Knight gombjának lenyomásakor kiválasztjuk a Knight-ot
* Eredmény: Sikeresen kiválasztottuk a Knight-ot
* Elvárt kimenet: Az Archer gombjának lenyomásakor kiválasztjuk az Archer-t
* Eredmény: Sikeresen kiválasztottuk a Archer-t

# Összegzés

A játék fejlesztése alatt Unity, C#, valamint a PHP ismereteinket bővítettük. Szívesen foglalkoznánk ezek után is játékfejlesztéssel és weboldalkészítéssel mivel nagyon élveztük az ezzel töltött időt. Sok érdekességet meg tudtunk a játék és a web készítése során, hogy mi hogyan működik.

# Forrásjegyzék

<https://hu.ign.com/hades/67008/feature/zsanermagyarazo-mi-a-fene-az-a-roguelike>

<https://www.makeuseof.com/platform-games-explained/>

<https://hu.ign.com/metroidvania/71403/news/zsanermagyarazo-mi-a-fene-az-a-metroidvania>

<https://docs.unity3d.com/Manual/system-requirements.html>

<https://unity.com/>

<https://visualstudio.microsoft.com/>

<https://jrsoftware.org/isinfo.php>

<https://free-game-assets.itch.io/free-fantasy-knight>

<https://darkpixel-kronovi.itch.io/archer-hero>

1. <https://hu.ign.com/hades/67008/feature/zsanermagyarazo-mi-a-fene-az-a-roguelike> [↑](#footnote-ref-1)
2. <https://www.makeuseof.com/platform-games-explained/> [↑](#footnote-ref-2)
3. <https://hu.ign.com/metroidvania/71403/news/zsanermagyarazo-mi-a-fene-az-a-metroidvania> [↑](#footnote-ref-3)
4. <https://docs.unity3d.com/Manual/system-requirements.html> [↑](#footnote-ref-4)
5. <https://unity.com/> [↑](#footnote-ref-5)
6. <https://visualstudio.microsoft.com/> [↑](#footnote-ref-6)
7. <https://jrsoftware.org/isinfo.php> [↑](#footnote-ref-7)
8. <https://free-game-assets.itch.io/free-fantasy-knight> [↑](#footnote-ref-8)
9. <https://darkpixel-kronovi.itch.io/archer-hero> [↑](#footnote-ref-9)