Tartalomjegyzék

3. A megjelenítő alrendszer modellje
    1. Statikus modell
        1. Kapcsolatok pontosítása
        2. Attribútumok azonosítása
        3. Bázisosztályok keresése
    2. Dinamikus modell
    3. Funkcionális modell
    4. Operációk azonosítása
    5. Az analízis modell osztálydiagramja.
    6. Az analízis modell osztályainak listája.
        1. Renderer
        2. Window
        3. DialogBox
        4. Scene
        5. SceneManager
        6. CombatScene
        7. MenuScene
        8. LevelScene
        9. InventoryScene
        10. Texture
        11. Spritesheet
        12. SpriteRenderer
        13. Sprite
4. A statisztikai alrendszer modellje
    1. Statikus modellek
        1. Kapcsolatok pontosítása
        2. Attribútumok azonosítása
        3. Bázisosztályok keresése
    2. Dinamikus modell
    3. Funkcionális modell

# 1. Vízió

## 1.1. Bevezetés

Szoftverünk neve: Ubimon
Csapatunk egy kétdimenziós RPG játékon dolgozik, amely néhány egyedi, és rendkívüli változtatások mellett nagy hangsúlyt fektett a környezetvédelem fontosságára.

## 1.2. Az alkalmazás helye

Az alkalmazás - elkészítése után - egy egyszerű, kis méretű RPG játékszoftver lesz, amely a Steam platformon és a Google Play áruházban lesz elérhető digitálisan, de tervben van egy fizikális, limitált kiadás is, mely tartalmazna kiegészítőket (kézikönyv, poszter, térkép stb.). A játék bármely korosztály számára ajánlott, ha valaki szeretne egy kicsit szórakozni, és mellesleg elgondolkodni a főtémával kapcsolatosan.

### 1.2.1. Üzleti lehetőségek

Ez a szoftver a szórakoztatóipart veszi célba. Egy fix áron Emellett számítunk arra, hogy a megfelelő balansz és stratégiák kialakítása végett, ez a játék eSport szintű bajnokságokba is bekerülhet.

### 1.2.2. A probléma megfogalmazása

Úgy véljük kevés fiatal ismeri miért is fontos a környezetvédelem, ezt a problémát szeretnénk azzal megoldani, hogy egy izgalmas és szerethető játék keretében megismertessük velük a mindennapi életben felmerülő lehetőségeket amikkel a felhasználó is hozzátehet a világunk megőrzéséhez.

Egy játékszoftveren belül – ha a verziókezelést figyelembe vesszük – igen nagy hangsúlyt kell fektetni a cél minél sokoldalúbb teljesítéséhez. Ehhez azonban érdemes a legalapabb problémákat megoldani, ahhoz hogy tovább tudjuk finomhangolni az univerzációt. Egy RPG-ben ezek a következők:

- Csalás nélküli végigjátszhatóság megoldása
- Megfelelő interfész kialakítása
- A játéktér beláthatósága
- Megfelelő AI biztosítása
- Változatos tér, és a szörnyek megfelelő elhelyezkedése
- Időzítések szinkronizálása, arányok bemérése
- A játékos cselekedeteinek végleges, vagy ideiglenes következménye legyen

**Csalás nélküli végigjászhatóság megoldása:**
Talán a legegyértelműbb probléma mind közül, mégis csak fontos számunkra ez a pont, főleg egy RPG-nél, ahol sokkal több lehetőség adott a játékos számára.

**Megfelelő interfész kialakítása:**
A műfajnak megfelelően a játék alatt két magasabb szintű interfészt tervezünk. Egy a harcrendszernek szánt felület (mely biztosítja a szörnyek harcba küldését, azok fejlődést, és az ellenfeleink legyőzését), emellett egy felülnézetes open-world képet kell biztosítanunk, hogy a játékos saját maga dönthessen az útvonaltervéről.

**A játéktér belátható:**
A kétdimenziós felülnézeti interfész alatt játszik inkább ez az elem nagy szerepet, ahol fontos hogy el tudjunk navigálni a célunkig. Rendes pályaszerkesztés, megfelelő színpaletta/színvilág, és más grafikai elemek szükséges ennek megalkotására, majd ezeket balanszolni kell.

**Megfelelő AI biztosítása:**
A fokozatos nehézségi szint megvalósítása a progresszió alatt, már egy alapkövetelmény szinte minden videójátéknál, így gondoskodnunk kell arról, hogy a játékos a történet haladtával, már sokkal kihívóbb, de legyőzhető ellenfelek ellen mérje össze tudását.

**Változatos tér, és a szörnyek megfelelő elhelyezkedése:**
Mivel a szörnyek egyediségét nem csak személyiségük, hanem különböző saját típusuk is meghatározza, így érdemes ezeknek az élőlényeknek a hozzájuk illő környezetet megalkotni, és ezekre a helyekre elhelyezni őket.

**Időzítések szinkronizálása, arányok bemérése:**
A nehézségnek megfelelően fontos bemérni az arányokat, és a fejlődési időtartalmakat össze kell hangolni.

**A játékos cselekedeteinek végleges, vagy ideiglenes következménye legyen:**
Nagy hangsúlyt szeretnénk fektetni a játékos egyedi döntéseire/cselekedeteire. Olyan a haladást befolyásoló elemekre gondolunk, mint NPC-k közötti viszonyfejlesztés, opcionális missziók elvállalása, és azok elvégzése, a játékos szörnyeinek törődése (harcképesítése, tanítása, etetése, mosdatása stb.), és persze a környezet épségének fenntartása. Mind ezen pontok odafigyelése, csak pozitív hatással jár, de ugyanakkor ezek elhanyagolása csak még nehezebbé teszi a kalandot.

## 1.2.3 Az elkészült termék helye

*A szoftver futtatható változata elsősorban digitális formában lesz elérhető, de lassan és biztosan egy fizikai limitált kiadást és biztosítunk. A szórakoztatóiparon belül korhatáros besorolás alatt nincsen minimum kor meghatározva, így a játék bárki számára igénybe vehető és alkalmazható.*

Egyes alternatíváktól annyiban eltér, hogy a szoftver egy egyszerű történet elmesélése mellett, sok egyéb elgondolkodtató, az életben is alkalmazható kérdést tesz fel. Egyedi térképpel és kis szörnyekkel rendelkezik a játék, amely a terv egyik főpillére. Nem konzol exkluzív játékot szeretnék fejleszteni, hisz nyitottak vagyunk sok más platformra is, és ez még az eladások számát is megnöveli.

# 1.3. Érintettek és felhasználók

A projekt fejlesztését KoviUbiSoft végzi. A megrendelő és konzulens Sátán Ádám.

## 1.3.1. Az érintettek összefoglalása

| Elnevezés | Leírás | Szerep |
|---|---|---|
| Megrendelő | Az alkalmazás elkészítését igényli | finanszírozza az alkalmazás fejlesztésének és fenntartásának költségeit |
| Fejlesztők | A játék elkészítéséért, illetve további fejlesztéséért felelősek | Elkészítik majd fejlesztik és karbantartják a játékot |

## 1.3.2.A felhasználók összefoglalása

| Elnevezés | Leírás | Illetékes |
|---|---|---|
| Grafikus | Grafikát ellenőrzi és utómunkákat végez | - |
| Tesztelő | A játékmenet hibáit keresi | tester |
| Admin | Felügyeli a csalás mentes játékmenetet többjátékos mód esetén | admin |
| Hangmérnök | A játék hangjain végez utómunkákat | Sound engineer |
| Játékos | Játszik a játékkal | player |

## 1.3.3 Felhasználói környezet

A játék PC platformon fog futni, a jövőben lehetséges Android verzió is. A játékot lehet egyjátékos, co-op és többjátékos módban játszani, ezért a felhasználók száma változó. Többjátékos mód esetén a maximális játékosok száma 32. A játék egyjátékos módú része 40 óra játékidőt tartalmaz, amely kibővíthető többjátékos módban.

## 1.3.4. Illetékesek adatai

A fejlesztői csapat:
KoviUbiSoft

Tóth József tothjozsef225@gmail.com

Simonyák János s_janos@outlook.com

| | |
|---|---|
| Nagy Róbert | rob.iskola1@gmail.com |
| Bartók-Balogh Gábor | demiteto.bartok@gmail.com |
| Hegedüs Gábor | hegedus.gabor91@gmail.com |


### 1.3.5. Felhasználók adatai

A szoftvert bárki használhatja, kortól és nemtől függetlenül.
A szoftver megrendelője és a fejlesztő csapat konzulense Sátán Ádám

## 1.4. A végtermék áttekintése

### 1.4.1 A termék kapcsolatai

A játék különböző komponensei kapcsolatban állnak egy adatbázissal, amely tárolja a játékos mentés fájljait, karaktereit, tárgyait és ubimonjait.

### 1.4.2. A termék használatának előnyei

| A játékos haszna (előnyei) | Az ezt támogató rendszer jellemző(k) |
|---|---|
| 40 óra alap játékidő | A játék egyjátékos módja 40 óra contentet tartalmaz. |
| Co-op mód | A játék egyjátékos részét maximum 4 másik emberrel lehet egyszerre játszani, a játék nehézségi szintje a játékosok számával növekszik. |
| Többjátékos mód | A játékot többjátékos módban is lehet játszani, maximum 32 emberrel. |
| Témája a környezetvédelem | Felhívja a figyelmet a környezetvédelemre. |

### 1.4.3. Feltételezések és függőségek

Feltételezések:

- A felhasználó rendelkezik Internetkapcsolattal, az online funkciók működéséhez.

- A felhasználó tisztában van az operációs rendszere kezelésével.

- Egy időben csak egy felhasználó használja a terméket.

Függőségek:

- A felhasználó rendelkezik egy OpenGl-t támogató videokártyával.

- Java Dependencies elérhetőek a szoftver megfelelő működéséhez

- Ha nincs elég hely egy teljes adatbázisra, akkor lehetőség van osztott adatbázis használatára.

### 1.4.4 Költségbecslés

A fejlesztési ideje: 1 év.

Fejlesztők száma: 5.

Fejlesztő eszközök: ~ 390,000Ft

- OpenJDK - ingyenes

- LWJGL - ingyenes

- IntelliJ IDEA Community- ingyenes

- Github - $21 / hónap / ember

- MySQL - Ingyenes

Fejlesztők: 21,000,000Ft

### 1.4.5. Installáció

A játékot Windows, Mac és Linux operációs rendszeren a Steam platformon lehet beszerezni. Androidos okostelefonoknál pedig a Google Play áruházból lehet letölteni.

## 1.5. A végtermék jellemzői, biztosított szolgáltatások

### 1.5.1. Az Ubimon alapötletének áttekintése

A projekt fő témája környezetünk és annak védelme. Ennek megfelelően a hangsúly a játék világában levő, annak egyensúlyát megtartó különleges képességekkel rendelkező állat- és növényfajokon (Ubimonokon) van a hangsúly. Rajtuk kívül jelen vannak még a velük együtt élő emberek is. Az embereknek meg kell tanulniuk szimbiózisban élni ezekkel az Ubimonokkal, mivel az erejük felelőtlen vagy akár önző kihasználása komoly természeti katasztrófákat okozhat.

A főhős – akit a játékos szabadon formázhat – fő célja az Ubimonok gyűjtése és tanulmányozása, dokumentálása. Ennek ellenére rákényszerülhet arra, hogy cselekedjen a fent említett katasztrófák megelőzése érdekében. Saját Ubimonjait használva áll szembe más, vad Ubimonokkal, vagy akár az őket rossz célra használó gonosztevőkkel.

### 1.5.2. Funkciók

A játék több órányi egyjátékos tartalmat biztosít a felhasználónak.

A játék egy nyílt, egységes világban játszódik. A játékos az eddig felfedezett területek között szabadon mozoghat. Előfordulhat, hogy egyes területek nem érhetőek el azonnal a játékos számára, és csak a játék folyamán, később nyílnak meg.

A játék harcrendszerére jellemző, hogy körökre osztott. A harcban részt vevő Ubimonok sebessége dönti el, hogy melyiknek érvényesül előbb a köre.

Többjátékos módban lehetősége van a játékosoknak egymással kooperatív módon biztonságban tartani a játék világát, Ubimonokat cserélni egymás között, vagy akár saját Ubimonjaik erejét megmérettetni különböző arénákban.

### 1.5.3. Egyéb szolgáltatások

A különböző online funkciók eléréséhez szükség lehet egy felhasználói fiókra, ami Steames kiadás esetén a felhasználó Steam fiókja is lehet. A játékos mentett állásai feltölthetőek a Steam Cloud szolgáltatásba, hogy más számítógépen is ott folytathassa a játékot, ahol azt előzőleg félbehagyta. Ugyanez a funkció Android platformon Google-fiókba történő mentéssel valósulna meg.

## 1.6. Korlátozások

### 1.6.1. Személyi számítógépek tekintetében

Személyi számítógépeken a projekt által felhasznált LWJGL keretrendszert figyelembe véve az egyetlen fontosabb rendszerkövetelmény egy OpenGL API-t támogató grafikus kártya. Pontosabban a software követelmény specifikációkban megtalálható a rendszerkövetelmény.

### 1.6.2. Mobil platformok tekintetében

Mobil platformokon egyéb szoftveres követelmények is vannak. Az LWJGL Android operációs rendszeren megköveteli a legalább 24-es API szintet, ami Android 7-el egyenlő.

### 1.6.3. Egyéb korlátozások

A játék futtatásának feltétele a végfelhasználói licencszerződés elfogadása. Az online funkciók használatba vételének feltétele a felhasználói feltételek és egy adatvédelmi nyilatkozat elfogadása. Az adatvédelmi nyilatkozatot el nem fogadó felhasználó nem csatlakozhat a program szervereihez, én nem tárolhatóak ott az adatai. Offline viszont továbbra is játszható marad a játék.

## 1.7. Minőségi elvárások

1. A csapat szempontjából a legfontosabb minőségi elvárás, hogy a játék a lehető legrövidebb időn belül a felhasználók/játékosok számára elérhető legyen a legkevesebb kockázati tényező kizárásával. A bugok és hibák a lehető legalacsonyabb százalékban forduljanak elő, illetve a játékmenet zökkenőmentes legyen az esetlegesen előforduló és ki nem küszöbölt problémáktól.

2. A játék elérhető legyen minél több felhasználó számára azaz optimalizációval minél szélesebb gép konfigurációra eljuttassuk így gyengébb hardverek is egyaránt gördülékenyen tudják futtatni programunkat.

## 1.8. Dokumentációkkal kapcsolatos követelmények

## 1.9. Kockázat lista

Jelenleg a legnagyobb akadálynak a projekt eddig még csak tervezési státuszban levő Android mobil operációs rendszerekre tervezett kiadása ígérkezik. Az LWJGL – habár lehetséges mobil platformra is készíteni vele játékot – asztali számítógépekre tervezett keretrendszer. A nehézséget főként az okozza, hogy Android platform-specifikus dokumentáció nem létezik, leszámítva néhány példaprojektet. Jelenleg eme probléma kiküszöbölésére még nincs megoldás a folyamatos tesztelésen kívül.

Továbbá a játék többjátékos szolgáltatásainál figyelni kell arra, hogy megfelelő képességű szervereket használjunk, azok túlterhelésének elkerülése érdekében. Fennállhat túlterheléses támadás lehetősége is, tehát biztosítani kell a DDOS támadások elleni megfelelő védelmet.

# 2. Software követelmény specifikáció

## 2.1. Bevezetés

A projektünk során átlagos, egyszerű, és egyben versenyképes játékosok számára egyaránt könnyen befogadható RPG szoftver megalkotására törekszünk. A recept alapjáraton nem egyedi, de a meglévő struktúrát megpróbáljuk bővíteni, és valamilyen szinten még egyedibbé alakítani.
Ebben a dokumentumban inkább technikai leírást szeretnénk rögzíteni a szoftver specifikációjáról, annak érdekében, hogy nagyobb áttekintést adjunk a rendszer által biztosítandó főbb szolgáltatásokról.

## 2.2. Áttekintés

Projectünk egy Single/Multiplayer alapú MMORPG típusú játékot valósít meg, amelyben
a fejlesztő csapat által megálmodott szörnyecskék és a világuk felfedezésén keresztül élhetjük át milyen a Ubimonok világa. Játékunkkal minden korosztályt szeretnénk megcélozni, de főként azok fogják élvezni a terméket, akik szeretik a kaland és a változatos világgal rendelkező alkotásokat. Szoftverünkkel bárki megismerkedhet a környezetvédelemmel, illetve megtapasztalhatja milyen egy kis kedvenccel törődni,
foglalkozni és nevelni.

A készítők által figyelembe vett legfontosabb funkciók:

- A felhasználó számára egy egyszerű, átlátható és könnyen kezelhető kezelőfelület kialakítása
- A játékhoz egy meséhez hű grafikát készíteni
- A játékélmény minden korosztály számára a lehető legjobb legyen
- A program megfelelő működése eltérő erősségű rendszereken is
- Teljes értékű játékmenet minden felhasználó számára

Ahhoz, hogy a játékot teljes értékűen kitudja használni a felhasználó szüksége van a játék irányításához szükséges eszközökre, azaz egérre és billentyűzetre, illetve ezeknek az eszközöknek a használatához elengedhetetlen tudásra. A program használata során semmilyen korlátozást nem kötünk ki. Minden felhasználó a saját elképzelése szerint játsszon és annyit amennyit jónak lát és kedve tartja.

**FELHÍVJUK FIGYELMÉT, HOGY A TÚLZOTT JÁTÉKHASZNÁLAT FÜGGŐSÉGET ÉS ROSSZ KÖZÉRZETET OKOZHAT!**
A játék használata során kérjük tartson legalább 1 óránként szünetet! Álljon fel számítógépe elől és pihenjen majd csak ezután folytassa a játékot.

A felhasználóval szemben elvárt követelmény:

- Rendelkezzen a szükséges eszközökkel a játékélmény teljes kihasználáshoz.
- Tisztában legyen az egér és billentyűzet megfelelő használatával.
- Felhasználó eszközének rendszere elégítse ki a ’Minimum rendszer követelmény’ kérését.
- Operációs rendszerek alapvető ismerete (Alkalmazás elindítása/bezárása).
- Jó problémamegoldó képesség és helyzetfelismerés.
- Többjátékos mód használatakor megfelelő viselkedés és magatartás

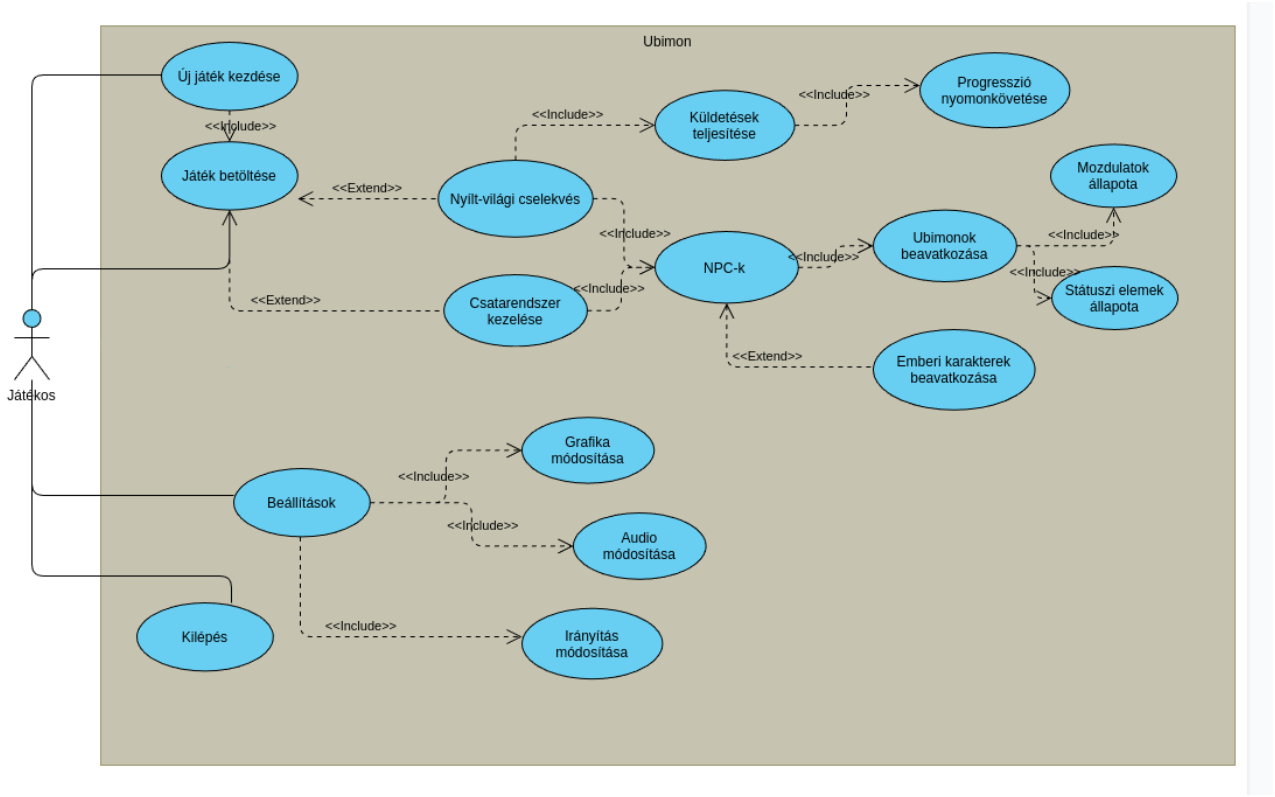
## 2.3. A rendszer funkciói

Az Ubimon alkalmazás használata során a felhasználó olyan funkciókat használhat, mint új/mentett játék elindítása, játékon belüli nyílt-világi tevékenység vagy az éppen csatában használatos interfész alkalmazása, játékon belüli feladatok teljesítése, Ubimonok gyüjtése és azok kiképzése, az emberi NPC-kel való viszony javítása, döntések meghozatala és még sok más.

Egy kisebb leírást adunk a szoftver főbb egységeiről:

- Menü: Egyszerű menü rendszer, amelyben a játékos új játékot kezdhet, folytathatja a meglévő save file-t, elérheti a beállításokat és kiléphet a játékból.
- Karakter kreáló: A karakter kreáló objektum felelős a játék elején azért, hogy a játékos létrehozza a karakterét.
- Ubimon trainer: A felhasználó által létrehozott karakter, amelyet a felhasználó irányíthat. Az ubimon trainerek rendelkeznek egy szinttel és tárgyakkal, amelyeket különböző szituációkban használhatnak.
- Ubimon labda: Az ubimon trainerek által birtokolt objektum, amely egy darab ubimont képes tárolni. A felhasználó a játék során szerzi, használja és menedzseli.
- Ubimon: Olyan objektum, amely a játék harcrendszerének alapvető eleme. Az ubimonokat a felhasználó birtokolhat ubimon labdákban. Az ubimonok rendelkeznek különböző tulajdonságokkal, pl.: típus, életpont, sebzés, stb.
- Óra: A játékban lévő valós idejű óra, amely a játékmenetet befolyásolja.
- NPC: Olyan karakter objektum, amelyet nem a felhasználó kezel. Az npc-k viselkedését a felhasználó játék során hozott döntései és eredményei befolyásolják. A játékos bizonyos npc-kkel képes beszélni, harcolni.

## 2.3.1. Offline egyjátékos mód




*Use Case diagram az alapvető felépítésről*

Szereplők közül itt még csak a ’Játékos’ szerepel, és a rendszer válaszaihoz más emberi beavatkozás nem csatlakoztatható.

- Új játék kezdése: E funkciót alkalmazva a felhasználó saját mentési slot-ot kiválasztva, a szoftver kezdeti állapotából indulhat.

- Játék betöltése: Akár új, akár elmentett állapotot betöltve elindul a szoftver főbb része, mely egy RPG esetén két nagyobb csoportú kezelőfelület adott kritériumoknak megfelelően váltogatja egymást.

- Nyílt világi cselekvés: RPG játékok esetén itt a játékos szabad döntéseket hoz, és próbálja a szoftver által kiadott küldetéseket teljesíteni, annak érdekében, hogy elérje a kaland végét. Azonban a szoftvernek nincs beleszólása a felhasználó választásain, így mindenki a saját döntése alapján növelheti a progressziót.-

- Küldetések teljesítése: Ennek az esetnek sok esetben teljesülnie kell az előrehaladás érdekében, de szerepjátékhoz híven itt biztosítunk mellékküldetéseket is, melyek kisebb mértékben növelik csak a progresszió értékét.

- Progresszió nyomon követése: A játék főküldetésének teljesítése, és a történet végén sem garantált, hogy a progresszió el fogja érni a maximumot, így fontos tudatni a felhasználóval, hogy hol tart, és vannak esetleges részei a játéknak, amelyek még befejezetlenek.

- Csatarendszer kezelése: A Nyílt világi cselekvés mellett, ez a második legfontosabb felület, amelynek protokollszerűen kell futnia. Ez az interfész ad képet, és biztosít a játékosnak irányítást arról, hogy körönként, milyen támadásokat vigyen véghez az Ubimonja.

- NPC-k: Itt csoportosul a különféle nem irányítható karakterek viszonya.

- Ubimonok beavatkozása: Normál esetben az Ubimonok engedelmeskednek a mestereinek utasítására, legfőképpen a csata alatt bizonyos támadások teljesítésére. Ugyanakkor open-world szinten ezek a szörnyek egyes helyeken még a környezetet is ápolják, és igénylik a természetnek való szolgálatot.

- Mozdulatok állapota: Egy Ubimon egyszerre csak négy mozdulatot tud alkalmazni, és ezeknek a mozdulatoknak a használata is korlátozottak. Vannak speciális mozdulatok is, melyek open-world tevékenykedés alatt is hasznosak, sőt elengedhetetlenek a továbbjutás érdekében.

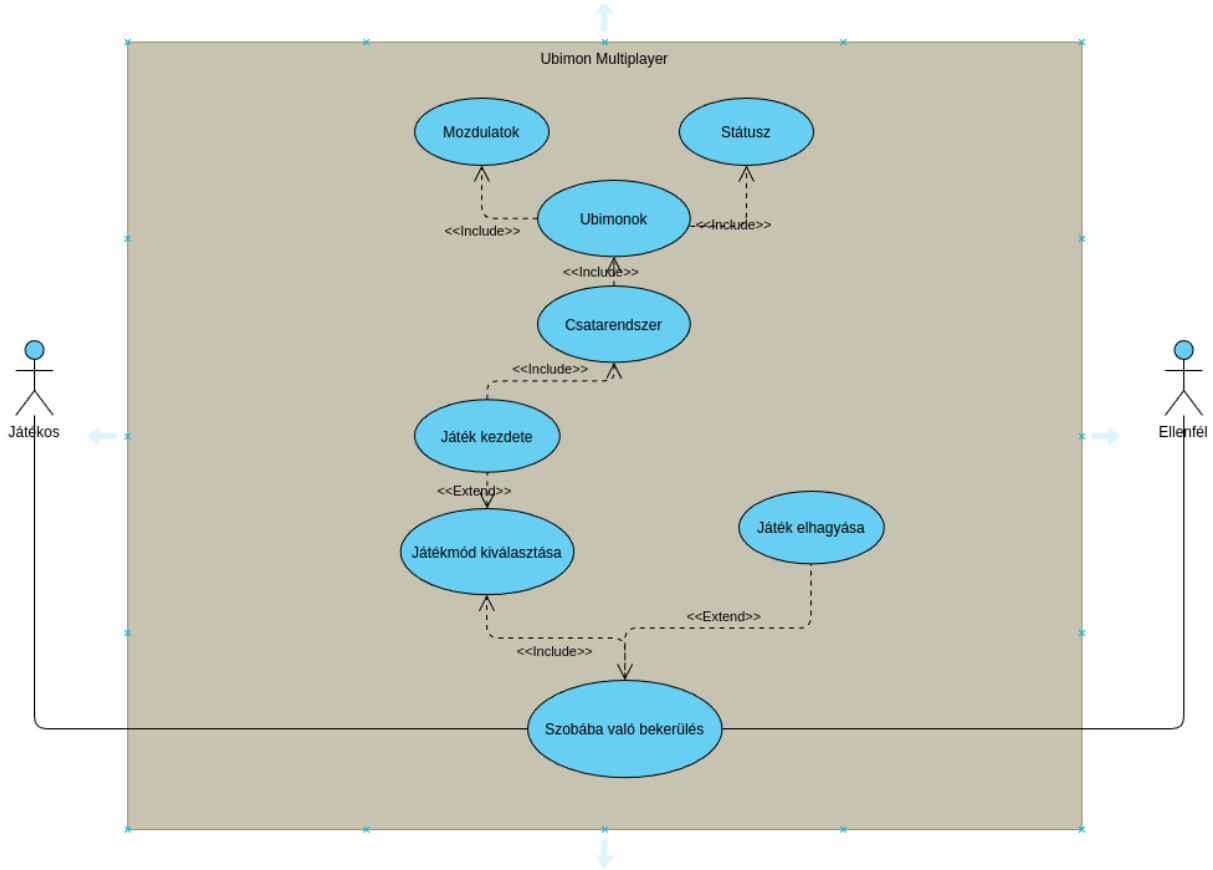
- Státuszi elemek állapota: Minden egyes Ubimon rendelkezik bizonyos státuszokkal (főleg Tamagotchi módban nő meg ezen elemek száma) melyeket érdemes figyelmet fordítani az eredményes harcok érdekében.

- Emberi karakterek beavatkozása: Eme fantasztikus világot rengeteg más ember is lakja, és több kategóriába sorolhatóak beavatkozásuk szempontjából. Vannak más edzők is, amelyek megpróbálják a játékost kihívni, és legyőzésük esetén több tapasztalati pont, illetve pénz jár. Emellett vannak átlagos emberi NPC-k is, amelyek vagy tanácsokat, tárgyakat, és más egyéb bónuszokat biztosítanak a felhasználó számára.

- Beállítások: Csoportosítja a testre szabható paramétereket, a felhasználó kényelmének. *Grafika, Audio, Irányítás*

- Kilépés: Ez a funkció biztosítja a szoftver legális bezárását.

## 2.3.2. Többjátékos mód




*Use Case diagram a többjátékos mód funkcióiról*

A ’Játékos’ szereplők mellett itt bekerül egy második szereplő is az ’Ellenfél’ néven, mely nagy vonalakban ugyanazokat a szerepeket tölti be, mint a ’Játékos’.

- Szobába való bekerülés: Többjátékos mód esetén itt különféle szobákat tartanak fent a szervereink, melyekben egyszerre csak két játékos tartózkodhat.

- Játék elhagyása: A szobában van lehetőség a játék megkezdése előtt elhagyni a szobát.

- Játékmód kiválasztása: Itt a nagyobb rangú felhasználó dönthet arról, milyen játékmódban mérjék össze tudásukat a játékosok. Emellett különböző profilokat is lehetőségünk van beállítani, annak érdekében, hogy ne keljen minden egyes alkalommal manuálisan minden paramétert újra és újra megváltoztatni.

- Játék kezdete: A megfelelő beállítások, és a „ready” rendszer szabályszerű lekérdezése után, a játék kezdődik, és itt már csak a csatarendszer felület lesz jelen az azt követő többi egységekkel együtt.

## 2.4. Használhatóság

***Kezelés elsajátítása:***

Fejlesztés során törekedtünk a könnyen átlátható és egyszerű játékmenet megteremtésére, a felhasználó által használt kezelőfelület letisztult és könnyen kezelhető kialakítást kapott ezáltal a játék rövid időn belül megtanulható és gond nélkül elsajátítható játékmenetet tartalmaz. Fontos megemlíteni, hogy a játék során előny lehet, ha a felhasználó játszott már hasonló logikát és játékmenetet tartalmazó játékkal, viszont ez nem szükséges a program kezelésének elsajátításában.

***Időigények:***

A kezelés megtanulásának egyszerűsége miatt a felhasználó számára minden adott, hogy mihamarabb elsajátítsa a játék hatékony használatát. Az elsajátítás időtartama függ a felhasználó képességétől és a játékmenet megtanulására fordított időtől.

***Rendszerszintű segítség:***

A szoftveren belül létrehoztunk egy „Sugó” nevezetű fület mely átfogó leírást ad a játék alapvető kezeléséről, ezáltal biztosítunk lehetőséget a felhasználó számára a játékon belül, hogy megfelelően és minél rövidebb időn belül elsajátítsa a program működésének alapelveit.

***Dokumentációk:***

A szoftver tartalmazni fog minden szükséges dokumentációt! A program használatához szükséges Felhasználói kézikönyv minden szükséges információt tartalmazni fog kezdő felhasználók számára. Tapasztaltabb játékosok ezek használata nélkül is könnyen meg fogják ismerni majd a játék lehetőségeit használat közben.

***Felhasználói felület:***

A kezelőfelületet a játék legcélszerűbb kezeléséhez alakítottuk ki. Nincsenek túlzó effektek vagy díszítések, csak a funkcionális dolgokat tartalmazza ezáltal eltér egy komolyabb játék kezelőfelületétől azonban egyszerűsége és átláthatósága biztosítja, hogy bármilyen képességű felhasználó pillanatok alatt megtanulja, hogyan kezelje a játékunkat.

***Futtatás:***

Szoftverünk futása semmilyen formában nem fogja befolyásolni más programok futását és használhatóságát! A program egy előre beállított ablakméretet fog használni. Alacsony gépigénye teljes mértékben lehetővé teszi más programok használata közbeni, megfelelő sebességű futtathatóságát. Minden akadály nélkül tudunk mellette internetezni vagy munkafolyamatot végezni anélkül hogy be kellene zárjuk alkalmazásunkat.

**FELHÍVJUK FIGYELMÉT, HOGY A MAGAS MEREVLEMEZ HASZNÁLATOT IGÉNYLŐ PROGRAMOK MINDEN ESETBEN LASSÍTJÁK MÁS SZOFTVEREK SEBESSÉGÉT!**

## 2.5. Megbízhatóság

- A szoftver rendelkezésre áll 100%-ban egyjátékos módban. Többjátékos mód esetén heti egyszer karbantartás előreláthatólag egy-két óra.
- MTBF (Mean Time Between Failures):  2186.5 óra
- MTTR (Mean Time To Repair): 4 óra
- Hibák kezelése: A program futás közbeni hiba esetén terminálódik.
- Hibák javítása: A javításokat a KoviUbiSoft fogja végezni.

## 2.6. Teljesítmény

- Válaszidők: Az egyjátékos mód esetén elhanyagolható adat. Többjátékos esetén európán belül 10 ms, 300ms fölött a kapcsolat megszakításra kerül amennyiben ez 10 másodpercen túl fennáll.
- Kapacitás: Indulásnál 10000 játékosra készülünk a többjátékos módban.
- Áteresztőképesség: 50000 TPS
    - Users (Threads): 10000
    - End to End Response Time: 1s
    - Pacing: 0s
    - Total Think Time: 0s
- Szerver:
    - CPU: Intel Xeon W-3300
    - RAM: 512GB DDR4 ECC
    - Videokártya: Aspeed AST2500
    - Tárhely:
        - 1TB Samsung 980 Pro PCI-E Gen4 M.2 SSD
        - 18TB Western Digital Ultrastar 7200 RPM Sata HDD
    - OS: Linux
- Erőforrás igények:
    - CPU: 2.2 GHz Dual core
    - RAM: 4 GB
    - OS: Windows 7 vagy újabb, Linux, macOS
    - Videokártya: Nvidia Geforce GTX 670 vagy AMD Radeon HD 7970 vagy újabb
    - Tárhely: 2 GB

## 2.7. Támogatottság

- A fejlesztők törekednek a Java nyelv szabványainak betartására, hogy a későbbiekben is könnyedén bővíthető kódot készítsenek.
- Az egyjátékos verzió nem igényel karbantartást. A többjátékos heti rendszerességgel történő karbantartását a KoviUbiSoft végzi.
- A hibabejelentésre külön lehetőség lesz a játék weboldalán.
- A szoftver előreláthatólag naplókat készít a hibákról és a bejelentkezésekről többjátékos mód esetén.

## 2.8. Tervezési korlátozások

A program fejlesztése Java nyelvben történik. Fontos számunkra, hogy a szoftver platformfüggetlen legyen, hogy minél több felhasználót el tudjunk érni. A nyelv további előnye az objektum orientáltság. Maga a fejlesztés a JetBrains IntelliJ IDEA Community fejlesztőkörnyezetében fog történni.

Célunk továbbá, hogy a játék rendszerkövetelménye a lehető legalacsonyabb legyen, ezzel biztosítva, hogy majdnem bármilyen hardveren futhasson.

Adatbázisnál a legfrissebb stabil MySQL verziót fogjuk igénybe venni.

## 2.9. Online dokumentáció és Help rendszer

A játék dokumentációja a játék weboldalán elérhető. A játék telepítése után a felhasználó számára lehető lesz egy offline súgó. A fizikai kiadás esetén felhasználói kézikönyv elérhető.

## 2.10. Felhasznált kész komponensek

LWJGL, ingyenes játék könyvtár. www.lwjgl.org. Licensz: BSD

## 2.11. Interfészek

### 2.11.1. Felhasználói interfészek

A program grafikus kezelőfelülettel (GUI) rendelkezik. A program grafikus elemekkel tart kapcsolatot a felhasználóval. Ilyen elemek például:

- Szövegdoboz: A játék ezeken keresztül ad információt a játékosnak, vagy írja ki a karakterek dialógusait.
- Választási lista: A játék időnként választási lehetőséget ad a játékosnak. Ezeket egy lista formájában jeleníti meg a játék, az opciókat egymás alá felsorolva. Példa erre a játékosnál levő tárgyak listája, egy választható válasz egy karakter kérdésére, vagy akár a harc közben használt menü is.
- Harc közbeni elemek: Ezek kommunikálják a játékos felé az éppen folyamatban levő harc állapotát, beleértve a saját és ellenséges Ubimon fajtáját, életerejét, szintjét, időjárási viszonyokat, stb.
- Szövegbeviteli mező: Lehetővé teszi a játékosnak, hogy szöveget gépeljen be egy szövegdobozba. Lehetséges billentyűzettel gépelni, vagy a szövegdoboz alatt található karakterek listájából sorban kiválasztani a megfelelő karaktereket.

Előfordulhat, hogy egyes esetekben nem jelenik meg kezelőfelületi elem, a játék világán és karakterein kívül. (pl.: jelenetek szöveg nélkül, vagy amikor éppen nincs szükség rá)

A játék kezelése a minimális kezelőfelületi elem miatt egyszerű, kezdő felhasználónak sem okoz különösebb problémát az elsajátítása.

### 2.11.2. Hardware interfészek

A szoftver támogatja a különböző játékvezérlő perifériákat, viszont nem igényli azokat. Az egyetlen igényelt hardver maga a számítógép/mobiltelefon, amin a program futni fog.

### 2.11.3. Software interfészek

A program futása egy támogatott operációs rendszert igényel, ami lehet Windows, macOS, Linux, vagy Android rendszer is. Személyi számítógépen szükséges a lehető legfrissebb Java Runtime Environment megléte.

### 2.11.3. Kommunikációs interfészek

A játék többjátékos módja aktív internetkapcsolatot igényel, amihez szükség van egy aktív kommunikációs csatornára. Ez a csatorna a felhasználói kliensek és a szerverek között van, ezen keresztül történik az adatátvitel. Ezt a csatornát bármilyen, a számítógépet az internethez csatlakoztató periféria biztosíthatja.

A távoli szerver eléréséhez TCP/IP protokollt használ a program.

A játékosok azonosítása PC platformon a Steamworks API segítségével történik. Minden játékosnak van egy egyedi SteamID-ja, aminek köszönhetően nem szükséges a játék szervereinek külön azonosítani a játékosokat, hiszen a Steam szerverei ezt már megteszik már a játék indítása előtt.

Mobil platformon az azonosítás Google Play fiókkal történik. Külön regisztrációra itt szintúgy nincs feltétlenül szükség.

Maga a játék szerverei az ezekhez az azonosítókhoz kötött adatok lekérdezéséért, feldolgozásáért, és a játékosok közötti interakciók feldolgozásáért szolgál.

## 2.12. Alkalmazott szabványok

Az alkalmazás fejlesztése alatt elengedhetetlen adott szabványok felhasználása. Itt megkülönböztettünk kötelezően és választás alapján alkalmazandó szabványokat. Kötelezően alkalmazandó szabványok alatt értjük azokat, amelyek kulcs fontosságú szerepük van a fejlesztés, és a termék legális kiadásának körében. A választás alapján alkalmazott szabványok alatt olyan elemeket soroltunk fel, melyek a csapat kényelmi döntés szerint lettek felhasználva.

### 2.12.1. Kötelezően alkalmazandó szabványok

Egy szabvány ami előre meghatározott a projekt fejlesztésében, az a Java nyelvre vonatkozó kódolási szabvány!
A kész játék életkor szerinti és tartalom szerinti besorolására az Egységes Európai Játékinformációs Rendszer, azaz a PEGI dönt.

### 2.12.2. Választás alapján alkalmazott szabványok

Alapjáraton 4:3-as képarány alkalmazását terveztük, de szélesvásznú kijelzők esetén (16:9, 16:10, 21:9) biztosítunk egy egyedi keretet az adott környezettől függően.

# 3. Rendszertervezés

## 3.1. Bevezetés

Az alábbi dokumentum tartalma a felhasználói felület vázlata grafikai elemekkel bővítve, valamint az adatbázis felépítése, beépülése a programba, és az azt kezelő modellek részletes leírása.

## 3.2. Felhasználói felület

A felhasználót a fő menü fogadja a játék elindulása után, ahol választhat, hogy új játékot kezd, betölt egy meglévő állást, többjátékos módot indít, a játék beállításait változtatja vagy kilép.


A beállítások menü négy részre osztott, ahol a játékos elérheti a hang beállításokat, grafikai beállításokat, irányítás és játékmenet beállításokat. A hang beállítások egyszerűek, lehetőséget adunk az effektek és a zene kikapcsolására valamit azok hangerejének precíz megadására. A grafikai beállításokban is törekedtünk egyszerűek maradni, felbontás, képarány, fényerősség, kontraszt és néhány fontosabb minőségi beállítás elérhető. Az irányítás beállításánál opció lesz a billentyűk testre szabása, valamint az egér érzékenységének beállítása. A játékmenetben a játék sebességét és a feliratokat tudjuk kezelni.


AUDIO
GRAPHICS
CONTROLS
GAMEPLAY
SOUND EFFECTS:
ON
SFX VOLUME:
BACKGROUND MUSIC:
ON
BGM VOLUME:
BACK


JÁNOS
LV. 18
HP:
TERRUIN
LV. 15
HP:
FIGHT
ITEMS
UBIMON
RUN

Harc közben látható interface jelzi majd a játékos, valamint az ellenfél életerejének a pontos állását, nevét és az ubimon szintjét egymáshoz közel, hogy ne kelljen a játékosnak keresgélnie ezeket a szűkséges információkat. A jobb alsó sarokban lesznek a harc közbeni lehetőségei.

Ezek a következők:

- Harc: itt lesznek az ubimon képességei és az alap támadása
- Tárgyak: harc közben használható tárgyak elérésére ad lehetőséget
- Ubimon: itt lehet váltani több ubimon között amennyiben a harca így jöttünk
- Menekülés: lehetőséget ad a játékos számára, hogy feladja a harcot

# 3.3. Adatmodellek

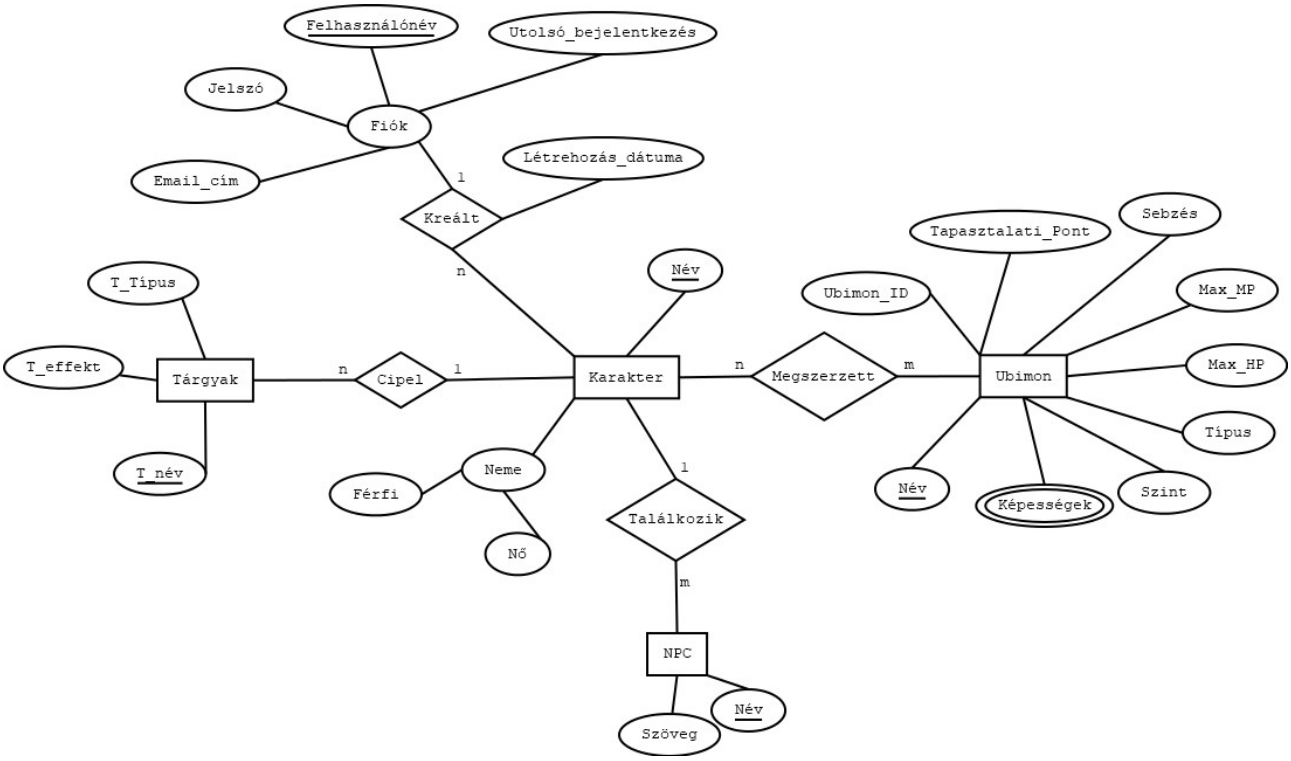
## 3.3.1. Adatbázis kezelő kiválasztása

A csapatunk a projekt költségvetésének szempontjából, ill. technikai okokból adatbázis kezelő modulként a MySQL program használata mellett döntött.

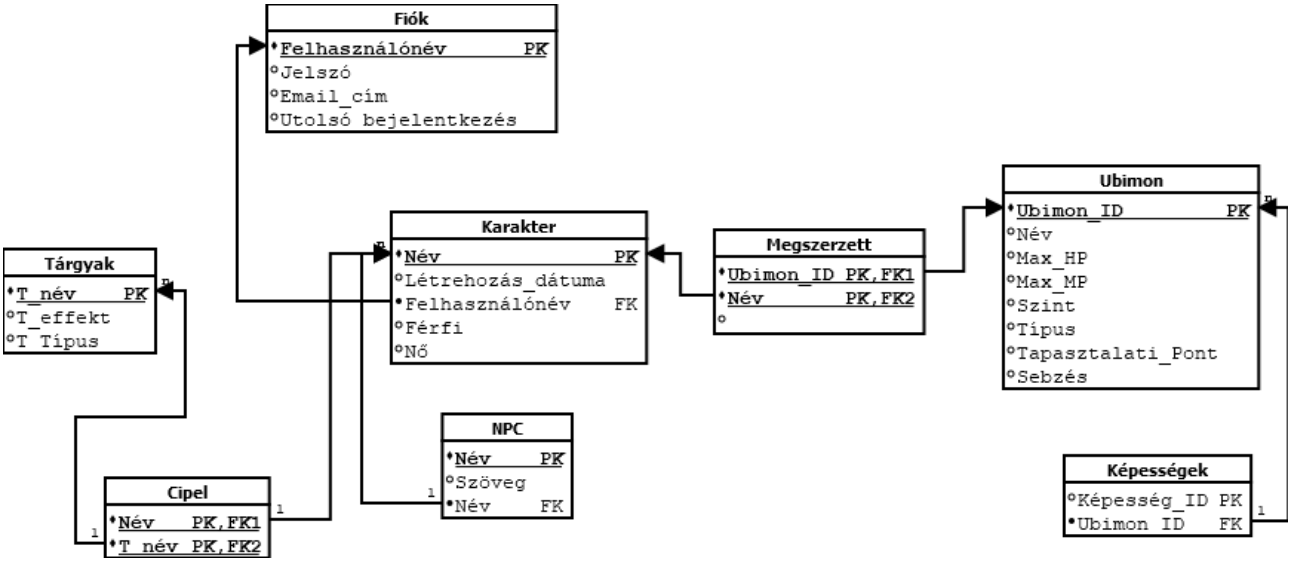
Indoklás:

- Ingyenesen letölthető és használható szoftver.
- Gyors adatbázis kezelést valósít meg, emellett egyszerü és könnyü használatot tesz lehetővé, így viszonylag könnyen megtanulható. Struktúrált lekérdező nyelv és akár önálló program írására is alkalmas.
- Újdonságaival, tudásában megközelíti a nagyobb teljesítményű adatbázis kezelő nyelveket, ugyanekkor megtartja a MySQL-től megszokott pozitív tulajdonságokat (gyorsaság, egyszerűség)
- A nézet egy egyszerű, de nagyon hatékony megoldás, melynek segítségével virtuális tábla szerű adatbázis objektumokat hozhatunk létre.
- A trigger lehetővé teszi, hogy az egyes adatbázis műveletek előtt és után valamilyen SQL kifejezés lefusson.
- Távoli táblák leképezését alkalmazza, ami valójában nem más, mint egy új adattároló motor, ami nem helyből, hanem egy másik adatbázisból olvassa az adatokat. Ezáltal hatékony lehet többgépes környezetben. Lényeges, hogy magas rendelkezésre állást biztosít , a nagy sebesség mellett.
- A lemezre írt műveleti adatok, illetve az időnkénti adatmentések segítségével gyors visszaállást is tud produkálni.

## 3.3.2. Szemantikai adatmodell




## 3.3.3 Relációs adatmodell




## 3.3.4. Az adatbázis kezelővel kapcsolatot tartó osztályok

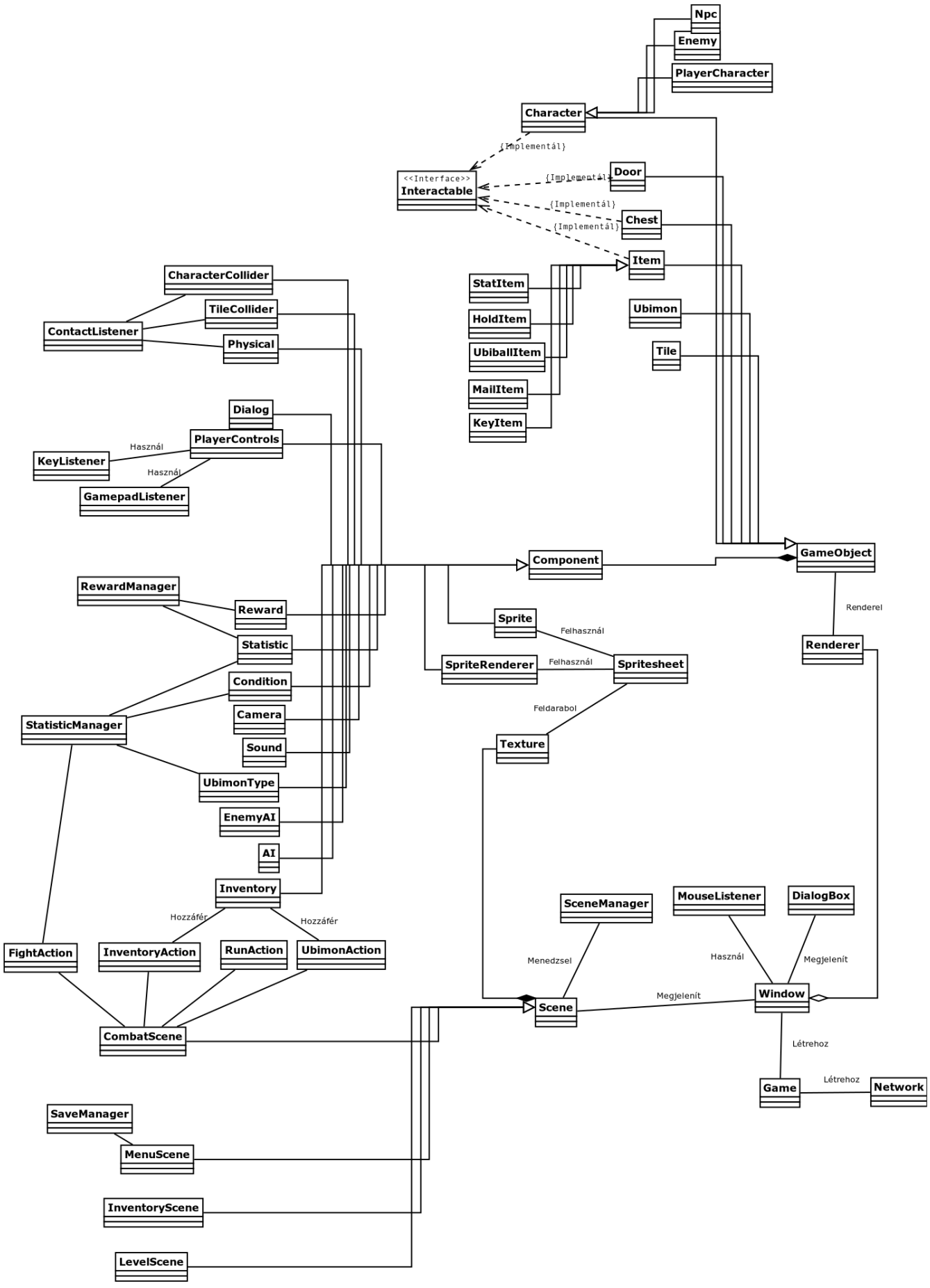
Az adatbázis kezelővel való kapcsolat tartást a Java programozási nyelv javax.sql csomag felhasználásával valósítjuk meg.

# 4. ANALÍZIS MODELL

## 4.1. Bevezetés

## 4.2. Kezdeti osztálydiagram

### 4.2.1 Osztálydiagram


Npc
Enemy
PlayerCharacter
Character
{Implementál}
<<Interface>>
Interactable
Door
{Implementál}
{Implementál}
Chest
{Implementál}
Item
CharacterCollider
TileCollider
ContactListener
Physical
StatItem
HoldItem
UbiballItem
MailItem
KeyItem
Ubimon
Tile
Dialog
PlayerControls
KeyListener
Használ
Használ
GamepadListener
Component
GameObject
RewardManager
Reward
Statistic
Condition
Camera
Sound
StatisticManager
UbimonType
EnemyAI
AI
Inventory
Sprite
Felhasznál
SpriteRenderer
Felhasznál
Spritesheet
Renderel
Renderer
Feldarabol
Texture
Hozzáfér
Hozzáfér
FightAction
InventoryAction
RunAction
UbimonAction
SceneManager
MouseListener
DialogBox
Menedzsel
Használ
Megjelenít
Megjelenít
Scene
Window
CombatScene
Létrehoz
Game
Létrehoz
Network
SaveManager
MenuScene
InventoryScene
LevelScene

## 4.2.2 Osztályok felsorolása

Component: Absztrakt osztály, amely egy adott GameObjecthez tartozik, tartalmazza a komponensehez kapcsolódó alapvető függvényeket és adatokat.

A Component absztakt osztályt kiterjesztő osztályok:

CharacterCollider: A karakterek ütközéséért felelős osztály.

TileCollider: A "csempékkel" történő ütközésért felelős osztály. (Hogy ne lehessen rajtuk áthaladni)

Dialog: A karakterek szövegéért felelős osztály.

PlayerControls: A játékos inputjához játékban történő történést rendel hozzá.

Reward: A karakterek által harc után kapott itemekért felelős osztály.

Statistic: A karakterek szintjéért, életerejéért, sebzéséért, stb felelős osztály.

Condition: Enum osztály, amely felsorolja a harcban szerezhető kondíciókat.

Camera: A játékost követő kamera, amely a játéktér "mozgatásáért" felelős.
gameobjecthez tartozó hangokért felelős.

Sound: A gameobjecthez tartozó hangokért felelős.

UbimonType: Enum osztály, amely felsorolja az ubimon típusokat.

EnemyAI: Az (nem játékos karakter) ellenség harcban használatos logikáját határozza meg.

AI: A nem játékos karakterek viselkedéséért felelős osztály.

Inventory: A karakter által bírtokolt tárgyakért és használatukért felelős osztály.

Sprite: A gameobject spritejának meghatározásáért és a hozzá tartozó függvényeket tartalmazza.

SpriteRenderer: Külön renderer a karakter gameobjecteknek, amelyeknek több spritejuk van. (animációhoz)

GameObject: Absztrakt osztály, amely a játékban található objektumokról fog információkat tárolni, mint pl.: név, komponensek listája, id.

A GameObjectet leszármaztató osztályok:

Door: Ajtó objektum, amely tartalmazza, hogy melyik kulcs item képes azt kinyitni. Implementálja az interactable interfészt. Hozzátartozó komponensek: Sprite, Sound, TileCollider.

Chest: Láda, amely itemeket tartalmaz. Implementálja az interactable interfészt. Hozzátartozó komponensek: Sprite, Sound, TileCollider.

Tile: Textúrával rendelkező "csempe" amelynek van Ütközése. Hozzátartozó komponensek: Sprite, TileCollider.

Ubimon: A harcban a karakterek által használt objektum, A játékos által használt ubimonok száma nem nagyobb, mint a játékos birtokában lévő ubimonlabdák száma. Hozzátartozó komponensek: Statistic, Condition, Sound, UbimonType.

Item: Az adatbázissal való kapcsolatért felelős (id alapján az item attributumait megszerzi). A játékos és a karakterek által bírtokolt tárgyakat és a hozzájuk tartozó függvényeket leíró osztály.

StatItem: Item osztályt kiterjesztő osztály, amely olyan itemeket tartalmaz, amelyek az ubimonok statisztikáját képesek befolyásolni.

HoldItem:  Item osztályt kiterjesztő osztály, amely olyan itemeket tartalmaz, amelyek a küldetésekhez kellenek.

UbiballItem:  Item osztályt kiterjesztő osztály, amely ubimon labdákat ír le (különböző effektek lehetnek).

MailItem: Item osztályt kiterjesztő osztály, amely szöveget tartalmaz.

KeyItem: Item osztályt kiterjesztő osztály, amely ajtókat vagy ládákat képes kinyitni.

Character: A Gameobject leszármazottja, amely a karakter alap adatait és függvényeit tartalmazza.

PlayerCharacter: A játékos által irányított karakter. A character osztály leszármazottja. Komponensei: CharacterCollider, Physical, PlayerControls, Dialog, Sound, Camera

NPC: A nem játékos által irányított karakterekre vonatkozó adatokat és funkciókat tartalmazza. A Character osztály leszármazottja. Komponensei: CharacterCollider, Physical, Dialog, Sound, AI, Inventory.

Enemy: Az ellenséges NPC-k adatait és függvényeit leíró osztály. A character osztály leszármazottja. Komponensei: CharacterCollider, Physical, Dialog, Sound, AI, EnemyAI, Inventory.

Renderer: A GameObjectek renderelséért felelős osztály

Window: Az ablakért felelős osztály. A lwjgl függvényeket használja.

KeyListener: Olyan osztály, amely a játékos billentyűlenyomásait figyeli, lwjgl segítségével.

MouseListener: Olyan osztály, amely a játékos egér inputjait figyeli, lwjgl segítségével.

GamepadListener: Olyan osztály, amely a játékos kontroller inputjait figyeli, lwjgl segítségével.

DialogBox: A karakterek dialóg szövegének kiíratásáért felelős osztály. Képes játékostól inputot kezelni.

Texture: A textúra fájlok adatait és kezelésének a függvényeit tartalmazó osztály.

Spritesheet: A textúrát spritokká feldaraboló osztály.

Scene: Absztrakt osztály, amely rendelkezik rendererrel, kirajzolja a háttér textúrát és a scenehez tartozó objektumokat.

LevelScene: A scene osztály leszármazottja.

CombatScene: A scene osztály leszármazottja.  A harcrendszer scene-je, amely tartalmazza a harchoz kapcsolódó játékos opciókat, megjeleníti az ubimonokat, azok statjait.

MenuScene: A scene osztály leszármazottja.  A játék menüje.

InventoryScene:A scene osztály leszármazottja.  A játékos inventoryjának az interface-e.

FightAction: A harcban történő játékos által kiadható támadásparancsok elvégzéséért felelős.

RunAction: A harcból való menekülésért felelős osztály.

InventoryAction: A harban az inventory használatáért felelős osztály.

UbimonAction: A harban történő ubimon cseréért felelős osztály.

SaveManager: A játékos által elmentett mentéseket és beállításokat betöltő, törlő és új mentéseket létrehozó osztály.

StatisticManager: A harcban, illetve item használata esetén az ubimonok statjait számítja ki, azok típusa, szintje és kondíciója alpján.

RewardManager: A harc utáni jutalomért feleős manager, amely a játékos és az ellenfél szintjét is figyelembe veszi.

Interactable: Interfész, amely az adott tile-el vagy karakterrel való interaktivitásért felelős.

ContactListener: A Physical és a tileok által meghatározott pozíció alapján az ütközésket figyelő objektum.

Networking: A hálózaton történő játékért felelős objektum

Game: A window-t és a Networking osztályokat használó (main) osztály.

### 4.2.3. Alrendszerek

Statisztika alrendszer:
Az ubimonok statisztikáit: életpontokat (hp), sebzést(dmg), kondíciót, szintet menedzselő alrendszer. Az ubimonok statjai harc közben vagy itemek használatakor változhatnak. Az alrendszer részei: StatisticManager, FightAction, Statistic, Condition, UbimonType.

Jutalom alrendszer:
A harc utáni jutalmi itemekért felelős alrendszer. Az alrendszerben résztvevő osztályok: RewardManager, Reward, Statistic

Harc alrendszer:
A harc helyzetben a játékos által választható opciók megvalósításáért felelős alrendszer. Az alrendszerhez tartozó osztályok: FightAction, InventoryAction, RunAction, UbimonAction.

Megjelenítő alrendszer:
A játékban megjelenő elemekért felelős alrendszer. Az alrendszerhez tartozó objektumok: Sprite, SpriteRenderer, Spritesheet, Texture. Window, Scene, SceneManager, CombatScene, MenuScene, InventoryScene, LevelScene.

Ütközés alrendszer:
A karakterek pozíciója és a tile-ok pozíciója alapján ütközést figyelő alrendszer. Részei: ContactListener, CharacterCollider, TileCollider, Physical.

Network alrendszer:
Az online többjátékos módért felelős alrendszer. Része: Network.

## 4.3. A megjelenítő alrendszer modellje

### 4.3.1 Statikus modell

#### 4.3.1.1 Kapcsolatok pontosítása

A megjelenítő alrendszer felel a játékban található összes grafikus elemének ábrázolására. A Renderer áll a grafikai megjelenítés feldolgozásának főpillérénél, amely közvetlen kapcsolatot ápol a GameObjecttel. A Renderer alatt lévő második ág a Window biztosítja a grafikai elemek egy nagy kereten belüli eltárolását, amely – a mi döntésünk alapján – mellesleg közvetlenül elérhetővé teszi a DialogBox-ot, hiszen a jelenetek minden egyes pontján nagy szerepet kap.
A Window után elérkezünk a többértékű Scene osztályhoz is, ami a leszármaztatott kisebb játékjelenetekkel kapcsolatos osztályokból alkot egy csoportot, melyek a szoftver adott pontján más jelenetek metódusaival dolgozik és tölti be.
A Texture és az az alatt lévő többi alosztály fontos szerepet játszanak a grafikai elemek beazonosítására, kirajzolására és azok tárolására szabályszerűen.

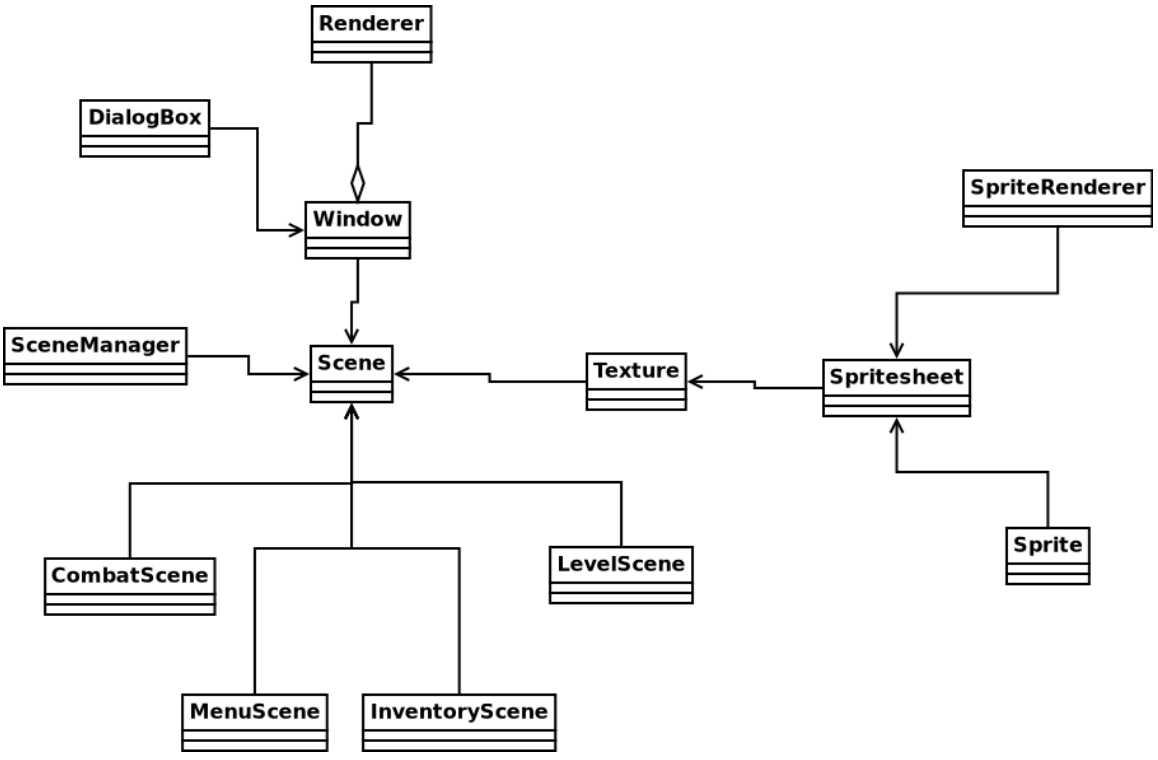
### 4.3.1.2 Attribútumok azonosítása

Az ebben az alrendszerben található osztályok többsége nem rendelkezik sok tulajdonsággal, mivel ezek feladata többnyire adott képsorozat (pálya) vagy objektum betöltése vagy rendelése. A képek célkoordinátáinak tárolására nincs szükség, mert azt mindig megkapják a kirajzolást kérő féltől. A Texture ősosztály leszármaztatottjai, azok amelyek a legtöbb aritmetikai számítást végzik el, többek között adott játék objektumok animációs fázisát a Spritesheet-ből.
A Scene osztály alatt található többi leszármazott osztály inkább csoportosítási és betöltési szerepet vállal.

### 4.3.1.3 Bázisosztályok keresése

Az összes grafikai osztály jellemző tulajdonsága, hogy tárolják a képet és gondoskodnak ennek megjelenítéséről, ezenkívül léteznie kell a képek betöltését végző metódusnak is, például Sprite-ok esetén a SpriteRenderer osztályt, amelyhez adott egy draw() nevű metódus, ami x,y koordinátákba kirajzolja a képet. A Sprite osztályban található adattagok például a MapSprite esetében tiszta a helyzet, egy képet tárol, amit meg kell tudni jelenítenie, azonban ObjectSprite-nál más a helyzet, hiszen valahogy tárolnia kell egy objektum különböző nézeteit és animációs fázisait, amit a Spritesheet tesz lehetővé.

## 4.3.2. Dinamikus modell



## 4.3.3. Funkcionális modell

Egyképernyő (pálya és egységek) kirajzolásának menete a következő:

1. Az adott scene-től függően a Scene egyenként meghívja a kirajzolni kívánt mezők draw() metódusát (pontosan közölve, hogy hova történjen a kirajzolás)

2. Miután a pálya kirajzolódott, a SceneManager üzenetet küld a látótérbe eső Objektumoknak, hogy rajzolják ki magukat

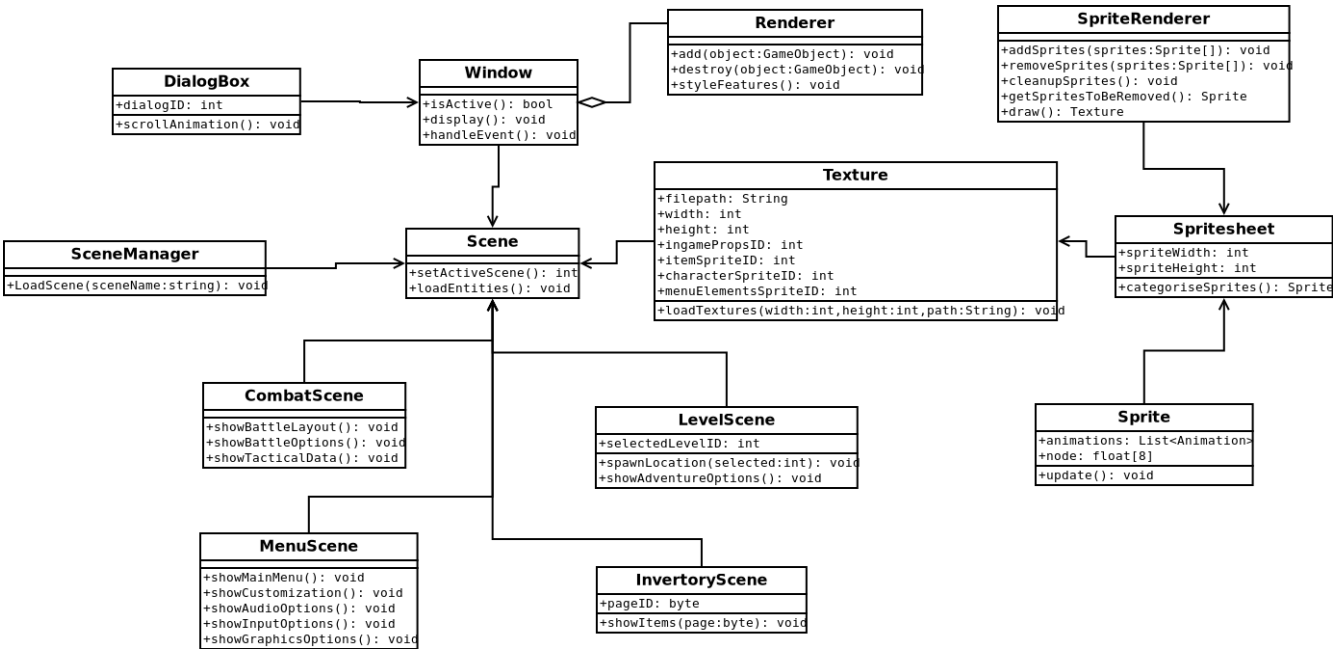
3. A SceneManager gondoskodik az animációról is, ezért bizonyos időközönként üzenetet küld a Scene-n belüli elemeknek, hogy frissítsék az animációs fázisukat.

4. Amikor a látótérbe eső egységek kirajzolására kerül a sor, a SceneManager üzenetet küld a SpriteRenderer-nek, hogy rajzolja ki a Scene-hez megfelelő Sprite-o(ka)t a Spritesheet-ből. Ekkor a draw() metódusa lefut a Renderer-ből, átadva neki a szükséges paramétereket.

## 4.3.4. Operációk azonosítása

A felhasználó a MenuScene láthatja, amikor elindítja a szoftvert egy a Window osztályon belül. A Scene-n belül lévő többi kisebb Scene osztály metódusai a játék bizonyos pontjain futnak le. A MenuScene választásait kattintható gombokkal tudja elérni, kontroller esetén a gombokkal, továbbá a menü irányítható a nyilakkal és az Enter, illetve Esc gombokkal. A játékmenet alatt szintén a nyilakkal irányítható a játszható karakter, a főkarakter további környezettel való interakciója pedig játékos által meghatározott gombbal történik főként klaviatúrán vagy kontrolleren.

## 4.3.5 Az analízis modell osztálydiagramja



## 4.3.6. Az analízis modell osztályainak listája

## 4.3.6.1 Renderer

Felelőssége, feladata: A renderer gondoskodik a játékban lévő összes objektum grafikai feldolgozásáról és azok megjelenítéséről.
Attribútumok: nincs
Operációk:

| Név | Típus | Feladat |
|---|---|---|
| add(object:GameObject) | void | Adott játék objektumok hozzáadásáért felel. |
| destroy(object:GameObject) | void | Adott játék objektumok eltűntetéséért felel. |
| styleFeatures() | void | Csoportosított grafikai elemek stílus funkció lekérdezésére szolgál. |

## 4.3.6.2. Window

Felelőssége, feladata: A játékszoftver által vetíteni kívánt értékek biztosítása egy ablakban.
Attribútumok: nincs
Operációk:

| Név | Típus | Feladat |
|---|---|---|
| isActive() | boolean | Igaz/hamis értéket visszaadó metódus idle vagy aktív állapot lekérdezése érdekében |
| display() | void | Szabályszerű megjelenítésért felelős |
| handleEvent() | void | Bizonyos események szabályszerű lekezeléséért felelős metódus |

## 4.3.6.3 DialogBox

Felelőssége, feladata: A játék bizonyos részein dialógus dobozokat jelenít meg, és ez az osztály gondoskodik az azonosításról, és a doboz animációjáról.
Attribútumok:

| Név | Típus | Leírás |
|---|---|---|
| dialogID | int | Bizonyos dialógusokat beazonosító ID. |

Operációk:

| Név | Típus | Feladat |
|---|---|---|
| scrollAnimation() | void | Szövegek scroll animációjáért felelős metódus. |

## 4.3.6.4. Scene

Felelőssége, feladata: A scene tárolja tulajdonképpen a játék minden egyes jelenetét, és a kisebb csoportokat a szoftver bizonyos pontján váltogatja és tölti be az azokhoz megfelelő egyedeket.
Attribútumok: nincs
Operációk:

| Név | Típus | Feladat |
|---|---|---|
| setActiveScene() | int | Ez a metódus felel adott kritériumoknak megfelelő jelenetek változtatására. |
| loadEntities() | void | A megfelelő jelenetekhez tartozó objektumok betöltésére szolgál. |

## 4.3.6.5. SceneManager

Felelőssége, feladata: A jelenetmenedzser segítségével tudunk a programban utalni a jelenetváltozásokra. Itt tudjuk kezdeményezni egy másik jelenet betöltését.
Attribútumok: nincs
Operációk:

| Név | Típus | Feladat |
|---|---|---|
| LoadScene(sceneName:string) | void | Ezt a függvényt hívjuk meg akkor, ha az adott feltételeknek megfelelően váltani szeretnénk egy másik jelenetre. |

## 4.3.6.6. CombatScene

Felelőssége, feladata: Az Ubimon játékban nagy hangsúlyt fektettünk az ellenfelek elleni megmérettetés alatt. Mint minden másik RPG-ben itt is fontos, hogy legyen egy külön harci jelenet, ahol a játékos dönthet arról, hogyan arasson győzelmet. Ez az osztály biztosítja e interfész megjelenítését.
Attribútumok:

| Név | Típus | Leírás |
|---|---|---|
| ubimonID | int | A játékos által bírtokolt ubimonID-je. |
| enemyUbimonID | Int | Az ellenfél által birtokolt ubimon ID-je. |

Operációk:

| Név | Típus | Feladat |
|---|---|---|
| showBattleLayout() | void | A harc jelenet alaprajzának megjelenítéséről gondoskodik. |
| showBattleOptions() | void | Biztosítja a felhasználó számára az elérhető lehetőségeket. |
| showTacticalData() | void | Taktikai/stratégiai információk elővetítésére szolgál. Sok kritérium változhat. |
| showUbimonStatistics(int:pu_id, int:eu_id) | void | A játékos és az ellenfél által használt ubimonok statjait megjelenítő függvény. |

## 4.3.6.7. MenuScene

Felelőssége, feladata: Ez a jelenet tartalmazza azon elemek megjelenítését, melyek a játék mechanikáját megvalósítják.
Attribútumok: nincs
Operációk:

| Név | Típus | Feladat |
|---|---|---|
| showMainMenu() | void | A játék főmenüjének a megjelenítéséről gondoskodik. |
| showCustomization() | void | A karakterkinézet választó jelenet megjelenítése új játék kezdése érdekében. |
| showAudioOptions() | void | Hang/zene beállítások megjelenítésére szolgál. |
| showInputOptions() | void | A bemeneti gombok beállítások megjelenítésére szolgál. |
| showGraphicsOptions() | void | Grafikai beállítások megjelenítésére szolgál. |

## 4.3.6.8. LevelScene

Felelőssége, feladata: Mivel a játékunk RPG műfajt alkalmaz, így a harc jelenet mellett, fontos biztosítani egy nyílt világot is, melyen a játékos saját karakterét irányítva szabadon dönthet az útvonaláról. Ez az osztály gondoskodik adott pálya résznek megfelelő betöltéséről és kezeléséről.
Attribútumok:

| Név | Típus | Leírás |
|---|---|---|
| selectedLevelID | int | Játék betöltése esetén szükség van adott pályarészek megfelelő azonosítására. Ezeket az azonosítókat itt tároljuk. |

Operációk:

| Név | Típus | Feladat |
|---|---|---|
| spawnLocation(selected:int) | void | Ez a metódus felel a selectedLevelID tulajdonságnak megfelelő pályarészhez való spawnolásához. |
| showAdventureOptions() | void | Bizonyos gomb aktiválása esetén előjön az ún. “kaland” beállítások, melyben elérhető a térkép, az invertory, a mentési opció, a statisztika, az Ubimonok stb. Ennek a menünek a megjelenítésére szolgál ez a metódus. |

## 4.3.6.9. InventoryScene

Felelőssége, feladata: A hátizsákunkban lévő tárgyak megjelenítésért felel.
Attribútumok:

| Név | Típus | Leírás |
|---|---|---|
| pageID | byte | Mivel adott tárgyak kategorizálva vannak, ezért ezeknek az oldalait külön azonosítanunk kell. |

Operációk:

| Név | Típus | Feladat |
|---|---|---|
| showItems(page:byte) | void | Bemeneti paraméternek megfelelő oldalon lévő tárgyak megjelenítése |

## 4.3.6.10. Texture

Felelőssége, feladata: Itt azonosítjuk be a játék alatt előforduló összes textúrának kategóriáját, fájl elérési helyét. magasságát, szélességét, illetve azokat töltjük be a renderelés alatt.
Attribútumok:

| Név | Típus | Leírás |
|---|---|---|
| filepath | String | A fájl elhelyezkedésének pontos címét tárolja. |
| width | int | Textúra szélességének értéke. |
| height | int | Textúra magasságának értéke. |
| ingamePropsID | int | A játék nyílt világ terén előforduló tereptárgyak azonosítására szolgáló érték. |
| itemSpriteID | int | Adott tárgyak beazonosítására szolgáló érték. |
| characterSpriteID | int | A karakterek spritejainak beazonosítására szolgáló érték. |
| menuElementsSpriteID | int | Adott menü grafikai elemek beazonosítására szolgáló érték. |

Operációk:

| Név | Típus | Feladat |
|---|---|---|
| loadTextures(width:int,height:int,path:String) | void | A jelenetnek megfelelő adott egyedekhez használandó textúrák betöltése. |

## 4.3.6.11. Spritesheet

Felelőssége, feladata: A spritesheet tartalmazza egy adott csoportban található elem több kisebb képekből álló állapotát, melyek a szoftver bizonyos részein töltődnek be, ezzel érzékeltetve az egyenletes animációt.
Attribútumok:

| Név | Típus | Leírás |
|---|---|---|
| spriteWidth | int | A sprite szélességének értéke |
| spriteHeight | int | A sprite magasságának értéke. |

Operációk:

| Név | Típus | Feladat |
|---|---|---|
| categoriseSprites() | Sprite | Bizonyos sprite elemek megfelelő csoportba való kategórizálásáért szolgál. |

## 4.3.6.12. SpriteRenderer

Felelőssége, feladata: A SpriteRenderer felel adott Sprite-ok megjelenítésére, és irányítja hogyan jelenjenek meg vizuálisan a jelenetek alatt.
Attribútumok: nincs
Operációk:

| Név | Típus | Feladat |
|---|---|---|
| addSprites(sprites:Sprite) | void | Sprite(ok) hozzáadása adott jelenethez a kritériumoknak megfelelően. |
| removeSprites(sprites:Sprite) | void | Sprite(ok) eltávolítása adott jelenetek alatt. |
| cleanupSprites() | void | Más jelenet beolvasása alatt érdemes váltás közben a többi nem aktív sprite-okat kitörölni optimalizáció érdekében. |
| getSpritesToBeRemoved() | Sprite | Egy olyan getter metódus, mely segít beazonosítani azon elemeket, melyekre bizonyos jelenetnél nincs szükségünk. |
| draw() | Texture | Ez a metódus szolgál a textúrák kirajzolására. |

## 4.3.6.12. Sprite

Felelőssége, feladata: A Sprite-ok kétdimenziós grafikai objektumok statikus textúrával, melyek bizonyos változások után váltakoznak általában lassabb képkocka/másodperces változással. Kevesebb számítást vesznek igénybe, mint a bonyolultabb 3D-s modellek. Ezek csoportosításával, azonosításával és frissítésével foglalkozunk ebben az osztályban.
Attribútumok:

| Név | Típus | Leírás |
|---|---|---|
| animations | List<Animation> | Animation típus alapú lista főként karakter sprite-ok esetén, hogy könyedén tudjuk kiolvasni adott pontokon lévő sprite cellákat. |
| node | float[8] | Ez a 8 elemű float tömb tárolja a sprite x y pontjait |

| | | |
|---|---|---|
| | | sorban. |

Operációk:

| Név | Típus | Feladat |
|---|---|---|
| update() | void | Egy folyamatos frissítésre szolgáló algoritmus, hogy minél gyorsabban tudjon a program sprite váltásokra reagálni. |

# 4.4. A statisztikai alrendszer modellje

## 4.4.1 Statikus modell

### 4.4.1.1 Kapcsolatok pontosítása

A statisztikai adatok lekérdezése egy távoli szerverről és egy lokális adatbázisból történik. Az programban egy osztály felelős azért, a fontosabb statisztikai elemeket lekérdezze a játék adott pontjain, és protokoll szerint menedzselje az értékváltozásokat. Az adatbázisból lehet lekérdezni az Ubimonok tulajdonságait, kondícióit, egyes játékbeli egyed statisztikáit.

### 4.4.1.2 Attribútumok azonosítása

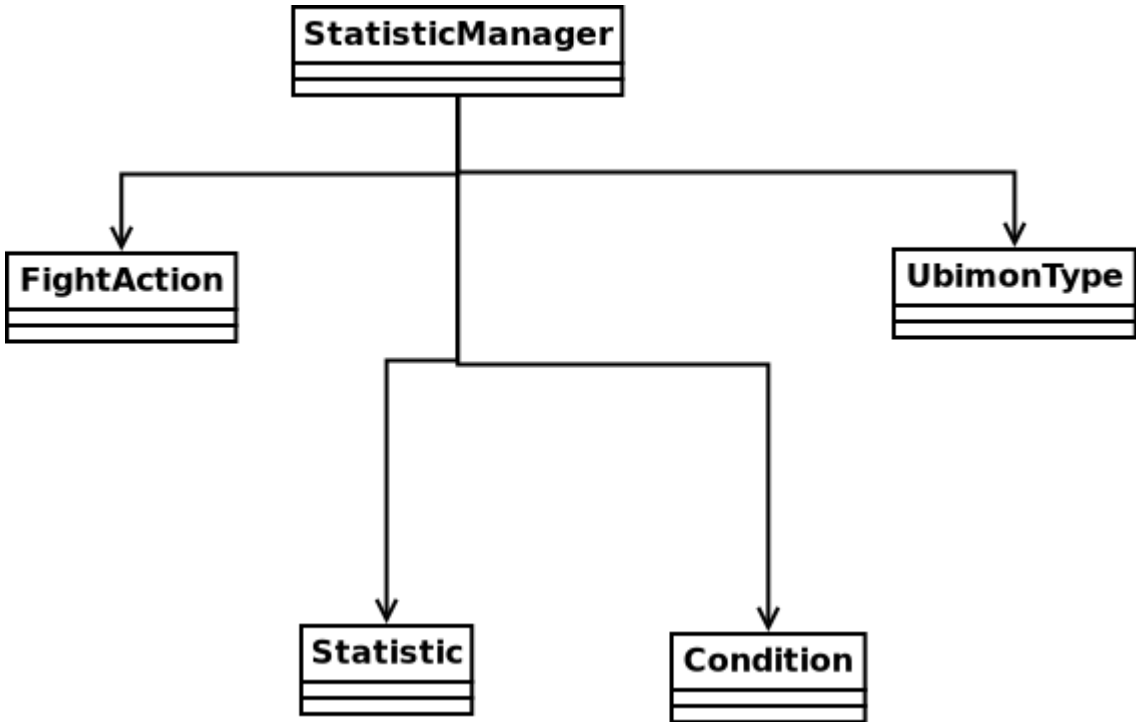
A statistic osztály legfontosabb attribútumai:

health: Az adott ubimon életpontja
exp: Az adott ubimon tapasztalatpontja
friendship: barátsági szint
level: Az adott ubimon szintje
Condition: A kondíciók listája
type: Az adott ubimon típusa
weakTypes: Azok az ubimontípusok, amelyre az adott ubimon gyenge.
strongTypes: Azok az ubimontípusok, amelyre az adott ubimon erős.

### 4.4.1.3 Bázisosztályok keresése

Nincs kifejezetten olyan osztály, amely bázisosztálynak kiemelhető lenne, hiszen az adatbázisba való írás, olvasás és lekérdezés műveleteket egy részét ez az alrendszer hajtja végre.

## 4.4.2. Dinamikus modell



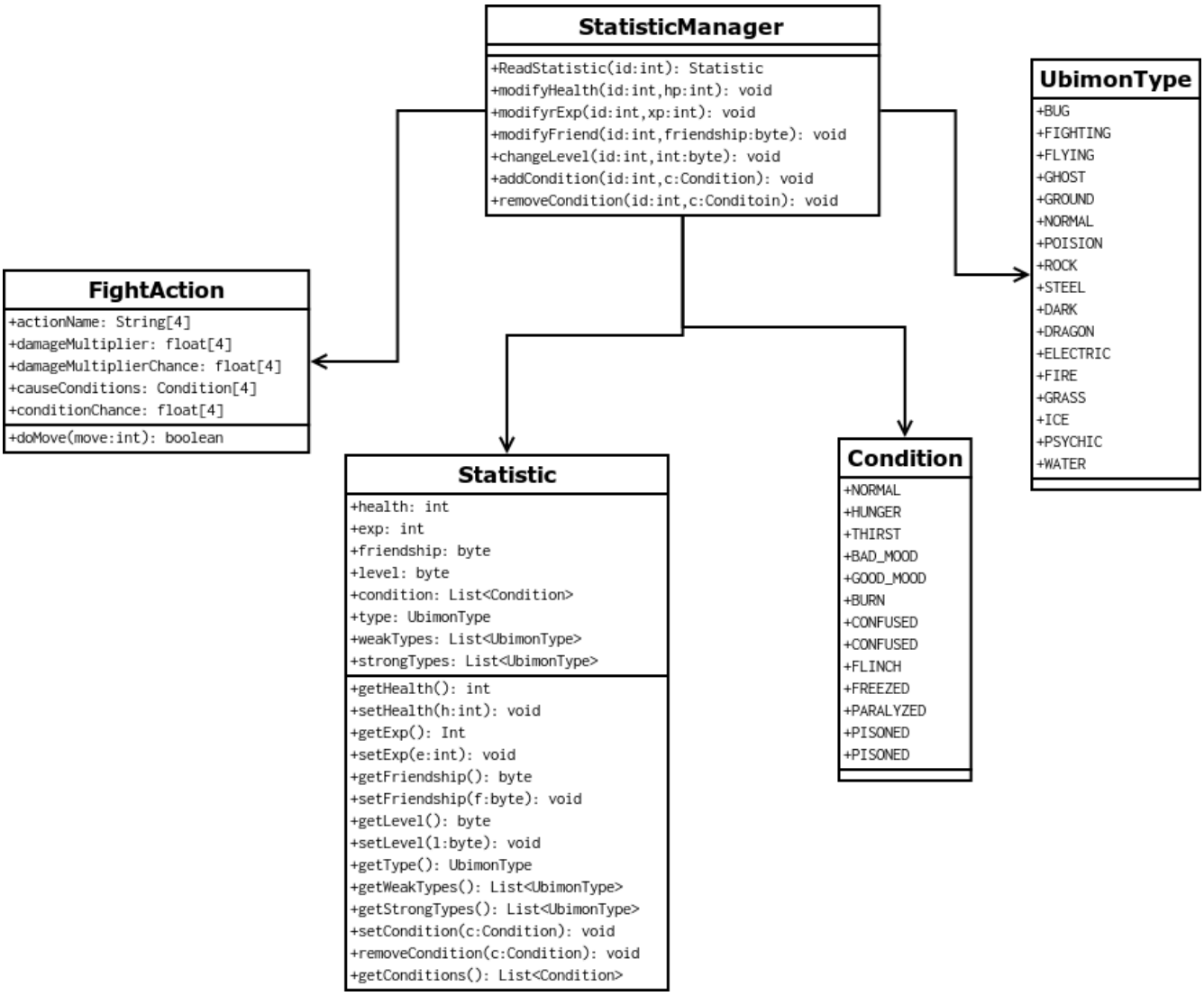

### 4.4.3.Funkcionális modell

A StatisticManager osztály ír és olvas az adatbázisból, módosítja az adatbázist a FightAction osztály üzeneteinek megfelelően, továbbá a GameObjectekhez tartozó Statistic osztály értékeit módosítja.

### 4.4.4. Operációk azonosítása

A játék alatt a statisztikai módosítások inkább a harcok alatt játszanak hatalmas szerepet (főleg ha a Tamagotchi mód ki van kapcsolva), hiszen a harcok alatt fejlődik a szörnyünk, kifáradhat, újabb mozdulatokat tanulhat meg ahogy nagyobb szinten lesz, státusz módosítokat rakhatnak rá stb. Ugyanakkor egyes kondíció értékek főként a nyílt világi tevékenységünk alapján módosul, hogy mennyire tartjuk az Ubimonunk éhség/szomjúság mértékén magasan, illetve mennyire gondoskudunk a kedvéről.

## 4.4.5 Az analízis modell osztálydiagramja




## 4.4.6. Az analízis modell osztályainak listája

## 4.4.6.1 StasticManager

Felelőssége, feladata: A statisztikai menedzser segítségével tudunk a programban utalni a statisztikai változásokra.
Attribútumok: nincs
Operációk:

| Név | Típus | Feladat |
|---|---|---|
| ReadStatistic(id:int) | Statistic | Ezt a függvényt hívjuk meg akkor, ha az adott feltételeknek megfelelően szeretnénk leolvasni adott Ubimon statisztikáit. |
| modifyHealth(id:int, hp:int) | void | A HP érték módosítása. |
| modifyrExp(id:int, xp:int) | void | Tapasztalatipont növelése/csökkentése. |

| | | |
|---|---|---|
| modifyFriend(id:int, friendship:byte) | void | Barátsági pont növelése/csökkentése. |
| changeLevel(id:int, level:byte) | void | A szint változtatása. |
| addCondition(id:int, Condition:c) | void | Kondíció hozzáadása |
| removeCondition(id:int, Conditoin:c) | void | Kondíció eltávolítása |

## 4.4.6.2. FightAction

Felelőssége, feladata: A harcban lévő Ubimonok statisztikai adatait, és cselekdeteinek metódusait tartalmazó osztály.
Attribútumok:

| Név | Típus | Leírás |
|---|---|---|
| actionName | String[4] | A támadások nevei |
| damageMultiplier | float[4] | A támadások sebzés szorzói |
| damageMultiplierChance | float[4] | A támadás sebzés szorzóinak valószíműsége |
| causeConditions | Condition[4] | Az okozható kondíciók |
| conditionChance | float[4] | Az okozható kondíciók valószínűsége |

Operációk:

| Név | Típus | Feladat |
|---|---|---|
| doMove(move:int) | boolean | A felhasználó által kiválasztott mozdulatot végzi el az Ubimon sok feltétel teljesülése után. |

## 4.4.6.3 Statistic

Felelőssége, feladata: Adott Ubimon statisztikáit tárolja, és metódusokkal módosítja.
Attribútumok:

| Név | Típus | Leírás |
|---|---|---|
| health | int | Az Ubimon életerje. |
| exp | int | Az Ubimon tapasztalatpontjai. |
| friendship | byte | Az Ubimon és a mestere közötti barátsági pont. |
| level | byte | Az Ubimon szintje. |
| condition | List<Condition> | Az Ubimon kondíciói. |
| type | UbimonType | Az Ubimon típusa |
| weakTypes | List<UbimonType> | Azok az ubimon típusok, amelyek extra sebzést okoznak az Ubimonra |
| strongTypes | List<UbimonType> | Azok az ubimon típusok, amelyekre extra sebzést képes okozni az Ubimon. |

Operációk:

| Név | Típus | Leírás |
|---|---|---|
| getHealth() | int | Életpont lekérdezése |
| setHealth(h:int) | void | Életpont beállítása |

| | | |
|---|---|---|
| getExp() | Int | Tapasztalatpont lekérdezése |
| setExp(e:int) | void | Tapasztalatpont beállítása |
| getFriendship() | byte | Barátsági pont lekérdezése |
| setFriendship(f:byte) | void | Barátsági pont beállítása |
| getLevel() | byte | Szint lekérdezése. |
| setLevel(l:byte) | void | Szint beállítása |
| getType() | UbimonType | Típus lekérdezése |
| getWeakTypes() | List<UbimonType> | Azok a típusok, amelyekre gyenge |
| getStrongTypes() | List<UbimonType> | Azok a típusok amelyekre erős |
| setCondition(c:Condition) | void | Kondíció beállítása |
| removeCondition(c:Condition) | void | Kondíció törlése |
| getConditions() | List<Condition> | Kondíciók lekérdezése |

## 4.4.6.4 Condition

Feladata: Adott Ubimon kondiciójának értékeit tárolja enum osztály.
Attribútumok:

| Név | Típus | Leírás |
|---|---|---|
| NORMAL | Condition | Egészséges státusz esetén ez az érték 0. |
| HUNGER | | Az Ubimon éhsége (Tamagotchi mód kikapcsolása esetén ez az érték lekérdezése nem aktív). |
| THIRST | | Az Ubimon szomjúsága (Tamagotchi mód kikapcsolása esetén ez az érték lekérdezése nem aktív). |
| BAD_MOOD | | Az Ubimon kedve (Tamagotchi mód kikapcsolása esetén ez az érték lekérdezése nem aktív). |
| GOOD_MOOD | | |
| BURN | | Minden körben elvesz az ubimon életéből, lecsökkenti a sebzést |
| CONFUSED | | 1 körben 50% esély, hogy az ubimon magát sebzi. |
| FAINTED | | Akkor érvényes, ha az ubimon élete 0 |
| FLINCH | | Nem támadhat 1 körig |
| FREEZED | | Nem támadhat, ameddig ki nem lesz olvasztva |
| PARALYZED | | A támadások 50% 0 sebzést okoz |
| PISONED | | Minden körben csökken az ubimon élete, a harc után is megmarad |
| SLEEP | | Pár körben nem támadhat. |

Operációk: Nincs.

## 4.4.6.4 UbimonType

Felelőssége, feladata: Ubimon azonosításra szolgáló osztály.
Attribútumok:

| Név | Típus | Leírás |
| --- | --- | --- |
| BUG | UbimonType | Ubimon típusok |
| FIGHTING | | |
| FLYING | | |
| GHOST | | |
| GROUND | | |
| NORMAL | | |
| POISION | | |
| ROCK | | |
| STEEL | | |
| DARK | | |
| DRAGON | | |
| ELECTRIC | | |
| FIRE | | |
| GRASS | | |
| ICE | | |
| PSYCHIC | | |
| WATER | | |

Operációk: Nincs

# 4.5. A harc alrendszere

## 4.5.1 Statikus modell

### 4.5.1.1 Kapcsolatok pontosítása

A harc alrendszer feladata, hogy meghatározza, hogy a játékosnak harchelyzetben milyen lépései lehetnek. A Scene osztályt leszármaztató CombatScene osztály fogja megjeleníteni a harcban résztvevő ubimonokat és a játékos által választható lehetőségeket, amelyek a FightAction, RunAction, InventoryAction és UbimonAction.
A FightAction a statisticManager osztállyal áll kapcsolatban, amely tartalmazza az adott ubimonhoz tartozó értékeket, kondíciót továbbá ezek típusát, amely meghatározza, hogy a harcban résztvevő pokémonok sebzési és védelmi értékei milyen szorzóval lesznek kiszámítva (2x, 1x, 0.5x, 0x). Az InventoryAction és az UbimonAciton kapcsolatban áll az Inventory osztállyal. Az inventoryAction kilistázza a játékos által bírtokolt olyan itemeket, amelyek a harcban használt ubimon statisztikáit módosíthatják (StatItem). Az UbimonAction kapcsolatban áll az Inventory osztállyal, mert az ubimonok Ubimon labdákhoz (UbiballItem) van kötve, az osztály lényege, hogy kilistázza a játékos által ubimon labdákban tárolt ubimonokat és harc közben, hogy a játékos tudjon ubimont váltani.

### 4.5.1.2 Attribútumok azonosítása

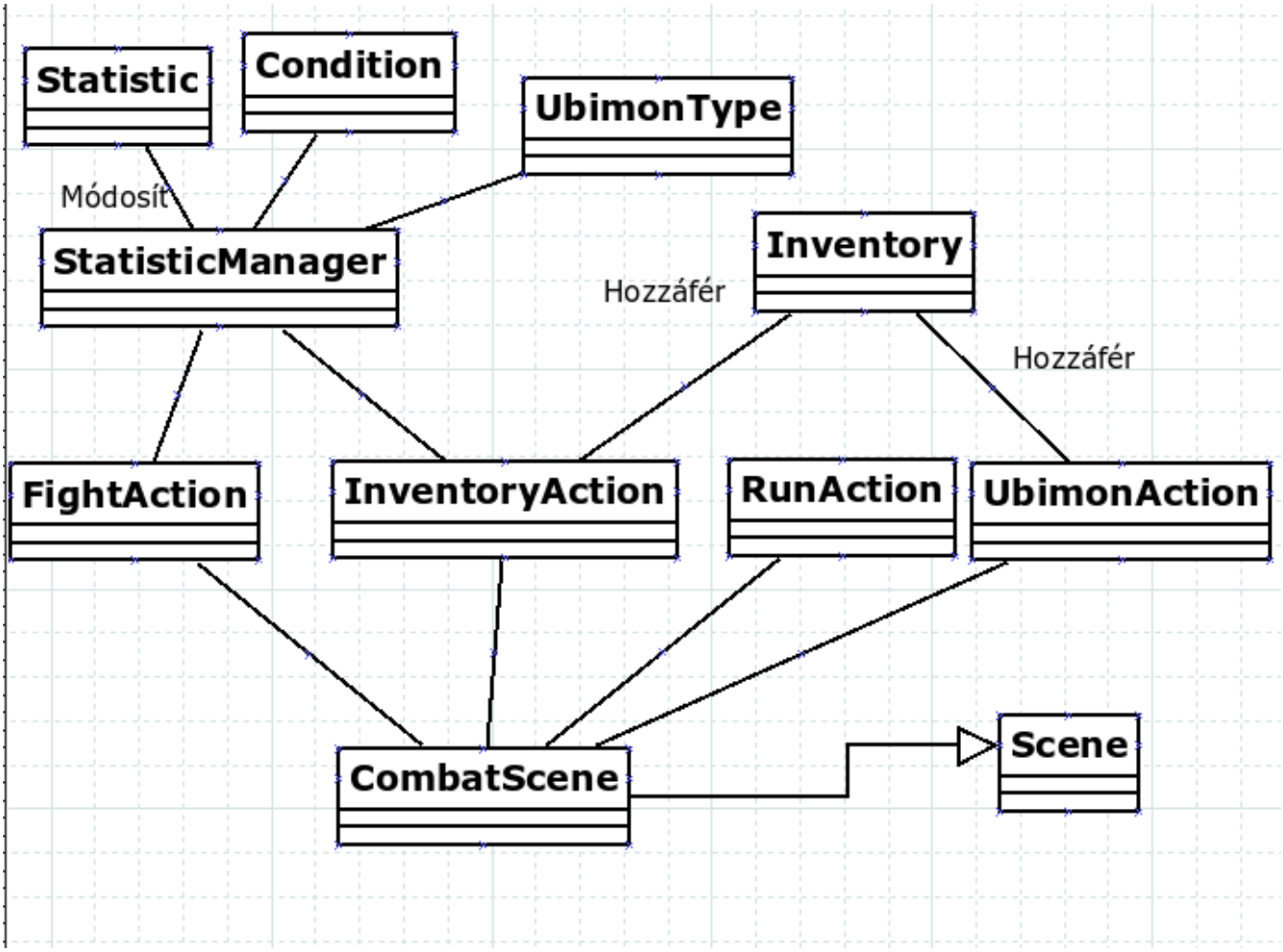
Az alrendszer legfontosabb attribútumai a FightAction osztályhoz tartoznak.
- actionName: A támadások neveit eltároló tömb.
- damageMultiplier: A sebzés szorzó értékeket eltároló tömb.
- damageMultiplierChance: A sebzés szorzó értékek valószínűségét tároló tömb.
- causeConditions: A támadás során létrehozható kondíciók listája.
- conditionChance: A támadás során kondíció létrehozásának a valószínűségét tároló tömb.

Az UbimonAction osztály attribútumja:
- currentID: A jelenlegi ubimon iD-je.

### 4.5.1.3 Bázisosztályok keresése

A harc rendszer osztályai nem rendelkeznek bázisosztályokkal (önálló osztályok).

### 4.5.2. Dinamikus modell




### 4.5.3 Funkcionális modell

Választás: A játékos választ a 4 lehetőség közül. (Fight, Run, Inventory, Ubimon)
Végrehajtás: A CombatScene osztály Inventory vagy Ubimon választás esetén megjeleníti a megfelelő menüt. Támadás kiválasztása esetén a megfelelő animáció kerül megjelenítésre. (Vizuális visszajelzések).
Jelzés a StatisticManagernek: A játékos támadás / sebződés vagy statisztikát befolyásoló tárgy használata esetén jelez a statisticManagernek, hogy változtassa meg az adott Ubimonhoz tartozó értékeket.
Ubimon váltás: Az ubimon spritjának és a sceneben megjelenített értékek kicserélése.

### 4.5.4 Operációk azonosítása

FightAction:
- doMove(move:int) A kiválasztott támadás végrehajtása
- activateHeld(item:byte) Kézben tartott item használata.

RunAction:
- run() Harc elhagyása.

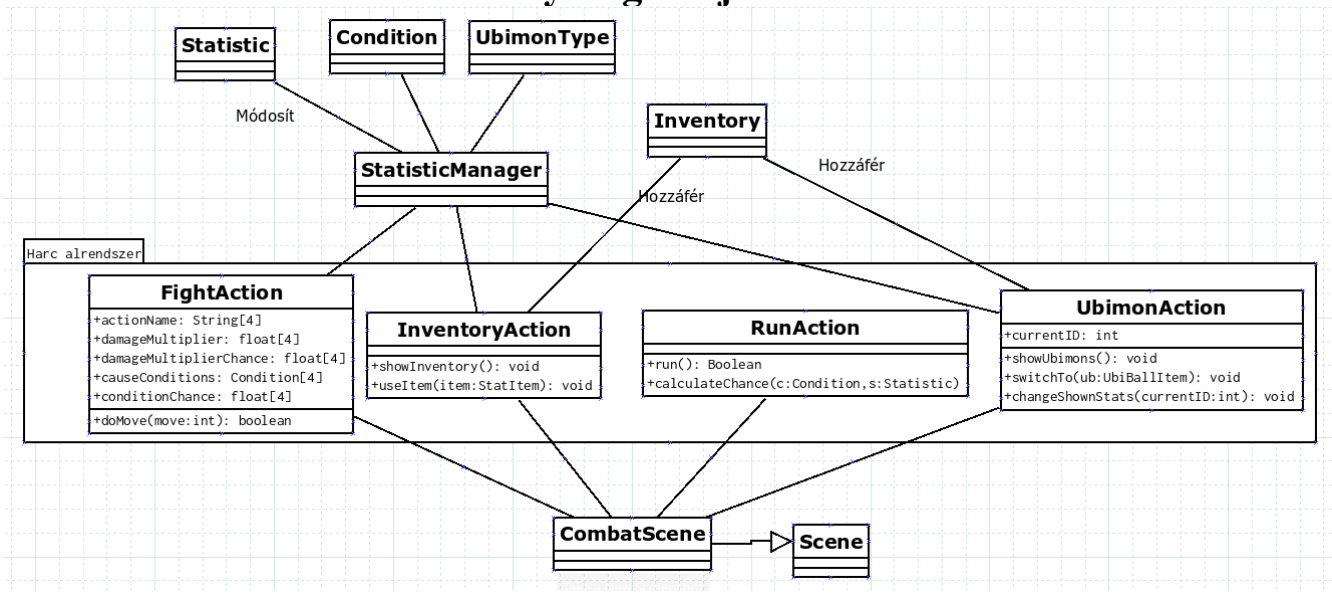
InventoryAction:
- showInventory() Inventory menü megjelenítése

- useItem(StatItem:item) Kiválasztott item aktiválása

UbimonAction:

- showUbimons() Ubimon menü megjelenítése
- switchTo(UbimonBall:ubimonball) Kiválasztott ubimonra váltás.

## 4.5.5 Az analízis modell osztálydiagramja




## 4.5.6.1. FightAction

Felelőssége, feladata: A harcban lévő Ubimonok statisztikai adatait, és cselekdeteinek metódusait tartalmazó osztály.
Attribútumok:

| Név | Típus | Leírás |
|---|---|---|
| actionName | String[4] | A támadások nevei |
| damageMultiplier | float[4] | A támadások sebzés szorzói |
| damageMultiplierChance | float[4] | A támadás sebzés szorzóinak valószíműsége |
| causeConditions | Condition[4] | Az okozható kondíciók |
| conditionChance | float[4] | Az okozható kondíciók valószínűsége |

Operációk:

| Név | Típus | Feladat |
|---|---|---|
| doMove(move:int) | boolean | A felhasználó által kiválasztott mozdulatot végzi el az Ubimon sok feltétel teljesülése után. |

## 4.5.6.2 InventoryAciton

Feladatai: A játékos által bírtokolt tárgyak kilistázása egy menüben. A játékos által kiválasztott tárgy aktiválása.
Attribútumok: Nincs
Opeációk:

| Név | Típus | Feladat |
|---|---|---|
| ShowInventory() | void | Az inventory menü megjelenítése. |
| useItem(StatItem:item) | void | Ubimon statisztikát módosító tárgy használata. |

## 4.5.6.3 RunAction

Feladatai: A harc elhagyása és a harc elhagyásának esélyét kiszámolni
Attibútumok: Nincs
Operációk:

| Név | Típus | Feladat |
|---|---|---|
| run() | Boolean | A harcből való eltávozás a feltételeknek megfelelően. |
| CalculateChance(Condition: c, Statistic: s) | float | A menekülés esélyét kiszámító függvény. |

## 4.5.6.4 UbimonAction

Feladata: Harc közben Ubimon váltás, megfelelő statisztika megjelenítése.
Attibútumok:

| Név | Típus | Feladat |
|---|---|---|
| currentID | int | A jelenlegi ubimonID-je. |

Operációk:

| Név | Típus | Feladat |
|---|---|---|
| showUbimons() | void | Az ubimonok listája. |
| switchTo(UbiBallItem:u) | void | Ubimon váltás. (id kicserélés) |
| changeShownStats(int:currentID) | void | megfelelő statisztikák megjelenítése |

# 5. Szótár

**1.1. Bevezetés:**
– RPG (Role Playing Game): Másnéven Szerepjáték: A játék programok világában egy olyan műfajnak a megnevezése, ahol egy általad létrehozott vagy előre meghatározott karakter bőrébe bújva egy adott szerepet játszol el, adott világon és adott történet szálon keresztül.

**1.2.2. A probléma megfogalmazása:**
– Open-world: Más néven nyílt világú: A játékban a szabadon bejárhatóságot jelképezi. Nincs kötött játék menet, kedved szerint derítheted fel a pályát.

– NPC: Más néven nem játszható karakter: A játékban egy játékos által nem irányítható karakter, amik támogatnak, vagy csak szimpla kiegészítés a játék világához.

– AI: Más néven Mesterségen Intelligencia: Egy gép, program vagy mesterségesen létrehozott tudat által megnyilvánuló intelligenciát nevezzük. A fogalmat legtöbbször a számítógépekkel társítjuk. Játékunkban a gép által kezelt NPC-k és Ubimonok AI által
vannak kontrollálva.

**1.2.3. Az elkészült termék helye:**
– exkluzív („Nem konzol exkluzív”): Csak egy adott platformon (PC, PS, Nintendo) elérhető alkotás. A mi játékunk nem fog rendelkezni ezzel a lehetőséggel.

**1.5.3 Egyéb szolgáltatások:**
– Steam Cloud: A Steam felhő alapú tárolója melyben a játék adatai fognak eltárolódni.

**1.6.3 Egyéb korlátozások:**
– Offline: Internet csatlakozás nélküli elérhetőség. A játék internet csatlakozás nélkül is játszható.

**1.7. Minőségi elvárások:**
– Bug: A játékban előforduló, elkészítés és kiadás után felmerülő apróbb problémák. A fejlesztők figyelmét elkerülő hibák, amely a tesztelés és a huzamosabb ideig tartó használat során derül ki.

**1.9 Korlátozások:**
– DDOS támadás: Olyan rendszer ellen irányuló terheléses támadás melynek célja a szolgáltatás elérhetetlenségének teljesítése.

**2.2.Áttekintés:**
– Single/Multiplayer: Egyjátékos és többjátékos mód a programban. Az egyjátékos mód offline internet kapcsolatot nem igénylő egyedül kijátszható játékmód még a többjátékos mód Online internet kapcsolatot igénylő más játékosok ellen játszható játékmód

**2.5.Megbízhatóság:**
– MTBF: Egy statisztikai alapú minőségi paraméter. Az elnevezés az angol nyelvű „Mean time between failures” kifejezés rövidítésből származik, mely a meghibásodások közt átlagosan eltelt időt jelenti.

– MTTR: Egy statisztikai alapú minőségi paraméter. Az elnevezés az angol nyelvű Mean Time To Repair kifejezés rövidítésből származik, mely azt mutatja meg, hogy a meghibásodott egység a meghibásodást követően mennyi idő múlva helyezhető ismét üzembe.

**2.6. Teljesítmény**

– ms: Többjátékos módban a szerver gép és a felhasználó eszköze közötti válaszidő mértékegysége (Milliszekundum)

– TPS: A tranzakció feldolgozás egyik módja. A tranzakció feldolgozó rendszer olyan szoftverrendszer vagy szoftver/hardver kombináció, amely támogatja a tranzakció feldolgozást.

– Users (Threads): Felhasználók, akik kapcsolódnak a szerverhez egy vagy több szálon.

– End to End Response Time: Az az időszak, ami eltelik a felhasználó által lenyomott gomb és a terminál válasz ideje között.

– Pacing: Az információk áramlását szabályozza a hálózaton belül, hogy elkerüljük a torlódást

– Total Think Time: Az eltelt idő egy kérés teljesítése és a következő kérés teljesítésének kezdése közt

– CPU: Az angol „Central Processing Unit” szóból származik a rövidítés magyarul Processzornak hívjuk. A számítógép „agya”, azon egysége, amely az utasítások értelmezését és végrehajtását vezérli,

– RAM: Az angol „Random Access Memory” szóból származik a rövidítés magyarul Memóriának nevezzük. Egy véletlen elérésű írható–olvasható adattároló eszköz. Amely a programok utasításait, adatait, a CPU munkájának eredményeit tárolja.

– TB: Informatikában használatos mértékegység „Terabájt”. A tárhelyünk méretének nagyságát jelzi (1TB = 1000GB)

– SSD: A merevlemezek egyik fajtája. A mozgó alkatrészek hiánya miatt kevésbé sérülékeny, mint a hagyományos merevlemez, hangtalan, kevés hőt termel, nincsenek a mechanikából adódó késleltetések, az adathozzáférés egyenletesen gyors. Gyorsabb, mint a HDD.

– HDD: A merevlemezek egyik fajtája. Az adatokat kettes számrendszerben, mágnesezhető réteggel bevont, forgó lemezeken tárolja. Lényegesen lassabb, mint egy SSD. Manapság inkább adattárolásra használják

– OS: Az angol „Operation System” szóból származik a rövidítés magyarul Operációs rendszer. Közvetlenül kezeli a hardvert, és egy egységes környezetet biztosít a számítógépen futtatandó alkalmazásoknak.

**2.11.3 Software interfészek:**

– Java Runtime Environment: Egy olyan szoftverréteg, amely a számítógép rendszerének a szoftverén fut, és biztosítja az osztálykönyvtárakat és egyéb erőforrásokat, amelyekre egy adott Java programnak szüksége van

– TCP/IP protokoll: Az internetet felépítő protokollstruktúrát takarja.

Alkalmazási réteg
Az alkalmazási réteg a felhasználó által indított program és a szállítási réteg között teremt kapcsolatot. Ha egy program hálózaton keresztül adatot szeretne küldeni, az alkalmazási réteg tovább küldi azt a szállítási rétegnek.
Szállítási réteg
Az alkalmazási rétegtől kapott adat elejére egy úgynevezett fejlécet (angolul: header) csatol, mely jelzi, hogy melyik szállítási rétegbeli protokollal (leggyakrabban TCP vagy UDP) küldik az adatot.
Hálózati (Internet) réteg
A szállítási rétegtől kapott header-adat pároshoz hozzáteszi a saját fejlécét, amely arról tartalmaz információt, hogy az adatot melyik végpont kapja majd meg.
Adatkapcsolati réteg
Az adatkapcsolati réteg szintén hozzárakja a kapott adathoz a saját fejlécét és az adatot keretekre bontja. Ha a kapott adat túl nagy ahhoz, hogy egy keretbe kerüljön, feldarabolja és az utolsó keret végére egy úgynevezett tail-t kapcsol, hogy a fogadó oldalon vissza lehessen állítani az eredeti adatot.

– SteamworksAPI: A Steam program egyik beépített rendszere melyen keresztül elérhetjük a Steamworks API által biztosított rendszereket. A játékunk esetében az azonosítást fogja kezelni

– SteamID: A SteamID egy egyedi azonosító a Steam programon belül melyet arra használnak, hogy azonosítsák a Steam fiókokat.

**3.2. Felhasználói felület:**
– New Game: Másnéven: Új játék: A felhasználó erre a szövegre kattintva kezdhet új játékba.

– Load Game: Másnéven: Játék betöltés: A felhasználó erre a szövegre kattintva elérheti a már elmentett játékmenetét.

– Multiplayer: Másnéven többjátékos mód: A felhasználó erre a szövegre kattintva csatlakozhat A játék többjátékos részlegéhez.

– Options: Másnéven beállítások: A felhasználó itt éri el a játékhoz kapcsolódó beállításokat Lásd a lentebbi képen.

– Quit: Másnéven kilépés: A felhasználó erre a szövegre kattintva kiléphet a programból.

– Audio: Másnéven Hang: A felhasználó itt éri el a játék hangbeállításait.

– Graphics: Másnéven Grafika: A felhasználó itt fogja tudni a játékhoz kapcsolódó grafikai beállításokat elvégezni.

– Controls: Másnéven irányítás: A felhasználó itt fogja tudni testre szabni a játék irányíthatóságát.

– Gameplay: Másnéven játékmenet: A felhasználó itt fog tudni a játékmenettel kapcsolatos beállításokat elvégezni.

– Back: Másnéven vissza: A felhasználó erre a gombra kattintva vissza tud lépni a főmenübe.

– Interface: Eszköz, vagy a számítógép és az azt használó ember érintkezési felülete. Az interfész olyan megoldásokat tételez fel, amelyeket mindkét fél ért.

# 6. Munkanapló

| Dátum | Verzió | Leírás | Szerző |
|---|---|---|---|
| 2021.09.24. | 0.1 | Prezentáció elkészülése | Tóth József, Bartók-Balogh Gábor, Hegedüs Gábor, Simonyák János, Nagy Róbert |
| **2021.09.28.** | **0.11** | **GitHub repository** | **Nagy Róbert** |
| 2021.10.01. | 0.15 | A feladatok szétosztása | Tóth József |
| **2021.10.05.** | **0.17** | **H2: 3. pont kitöltése** | **Hegedüs Gábor** |
| 2021.10.05. | 0.18 | H2: 4. pont kitöltése | Nagy Róbert |
| **2021.10.08.** | **0.2** | **H2: 1.-2. pont kitöltése** | **Tóth József** |
| 2021.10.08. | 0.25 | H2: 5. 6. és 9. pont kitöltése | Simonyák János |
| **2021.10.08.** | **0.3** | **H2: 7. pont és a Szótár kitöltése** | **Bartók-Balogh Gábor** |
| *2021.10.15.* | *0.31* | *H3 feladat kiosztás* | *Tóth József* |
| **2021.10.20.** | **0.35** | **H3: 9. és 10. pontok kitöltése** | **Nagy Róbert** |
| 2021.10.21. | 0.38 | H3: 1. és 12. pontok kitöltése | Tóth József |
| **2021.10.22.** | **0.4** | **H3: 8. és 11. pontok kitöltése** | **Simonyák János** |
| 2021.10.22. | 0.41 | H3: 9-es pont kiegészítése | Nagy Róbert |
| **2021.10.22.** | **0.43** | **H3: 5. és 6. pontok kitöltése** | **Hegedüs Gábor** |
| 2021.10.22. | 0.45 | Kézikönyv elkészítése | Tóth József |
| **2021.10.23.** | **0.47** | **Szótár frissítése** | **Bartók-Balogh Gábor** |
| 2021.10.23. | 0.5 | H3: 2. és 4. pontok kitöltése | Bartók-Balogh Gábor |
| **2021.10.24.** | **0.51** | **H3: 3-as pont egy részének kitöltése** | **Nagy Róbert** |
| 2021.10.24. | 0.53 | H3: 3-as ponthoz Use Case diagram | Tóth József |
| **2021.10.29.** | **0.54** | **H3: 11.4 pont bővült** | **Simonyák János** |
| *2021.10.29.* | *0.55* | *H4 feladat kiosztás* | *Tóth József* |
| **2021.10.29.** | **0.57** | **RendszerTervezés: 2-es pont rajzok** | **Tóth József, Simonyák János** |
| 2021.11.08. | 0.6 | Szótár frissítése | Bartók-Balogh Gábor |
| **2021.11.08.** | **0.62** | **Analízis dokumentumhoz osztály diagram** | **Nagy Róbert** |
| 2021.11.09. | 0.65 | RendszerTervezés: 2-es pont szöveg | Hegedüs Gábor |
| **2021.11.09.** | **0.7** | **RendszerTervezés: 3-as pont szöveg és Szótár** | **Bartók-Balogh Gábor** |
| 2021.11.09. | 0.75 | Analízis dokumentum: Harc, Grafika és Statisztika alrendszer | Tóth József, Simonyák János, Nagy Róbert |
| **2021.11.16.** | **0.77** | **RendszerTervezés: Forma javítás** | **Bartók-Balogh Gábor** |
| 2021.11.16. | 0.78 | H4 prezentáció | Hegedüs Gábor |
| **2021.11.16.** | **0.8** | **Analízis dokumentum: 4-es pont javítás** | **Nagy Róbert** |
| 2021.11.23. | 0.82 | Teszt v1 | Hegedüs Gábor |
| **2021.11.23.** | **0.84** | **Teszt v2** | **Nagy Róbert** |
| 2021.11.25. | 0.88 | Teszt v3 | Bartók-Balogh Gábor |
| **2021.11.25.** | **0.9** | **H5 dokumentum első verzió** | **Hegedüs Gábor** |
| 2021.11.25. | 0.94 | Teszt v4-v4.1 | Tóth József |
| **2021.11.30.** | **0.98** | **Teszt v5-v5.1** | **Simonyák János** |
| 2021.11.30. | 1,0 | H5 dokumentum kész | Hegedüs Gábor |