**GRAFIKUS FELÜLET SPECIFIKÁICÓJA**

40 – ZETA

Konzulens:

KOVÁCS BOLDIZSÁR

Csapattagok

|  |  |  |
| --- | --- | --- |
| Alpek Dávid Zsolt | C31X0F | alpek.david.zsolt@gmail.com |
| Csia Klaudia Kitti | HA5YCV | kitkat@sch.bme.hu |
| Litavecz Marcell | IPHJNB | marcell.litavecz@gmail.com |
| Marton Judit | M0MYIM | judit.marton@edu.bme.hu |
| Ruskó Eszter | H8IBRC | eszter@rusko.hu |

2022. 05. 03.

# Grafikus felület specifikációja

## A grafikus interfész

### Karakter képek

Itt azok a képek találhatóak meg, amelyek több játékos esetén rajzolja ki a különböző színű karaktereket, illetve viselt eszközöknél, hogy fog kinézni a *CharacterView-*ban a karakter, és bónuszként, hogy mivé fog alakulni egy medvetánccal fertőzött karakter. A zsák nem fog látszódni egy ábrán se, ugyanis az nálunk úgy van lekezelve, hogy csak plusz számláló bővítőként van hozzáadva a karakter *Inventory*-jához, így ahhoz nem készült rajz.



Az alap karakterünk, ez lesz a kiinduló verzió.



Ha több játékos is belekerül, akkor ilyen színnel fognak megjelenni a pályán.

A képen szöveg, játék, vektorgrafika, clipart látható

Automatikusan generált leírás

Alap karakterünk [balról jobbra] kesztyűvel, fejszével, majd kesztyűvel és fejszével. Ezek a verziók + a köpenyes verziók nem fognak látszódni az alap játékban, hanem kizárólag csak a *CharacterView*-ban, ugyanis alapból azt feltételezzük, hogy a karakterek vakok, így mindig csak az alap ruhát látják egymáson.



A karakterünk [balról jobbra] köpennyel, köpennyel és fejszével, köpennyel és kesztyűvel majd a legvégén mind a három eszközzel együtt.



Így néz ki egy medvetánccal fertőzött játékos, természetesen ez csak mások szemében néz ki így, ugyanis, akit megfertőztek az már csak a Game Over feliratot fogja látni. Másik játékosok szemében azonban egy folyamatosan mozgó, raktárt fosztó, véres-szemű, fertőző medvét fog látni, aki, ha megjelenik egy pályán számolhatunk a problémákkal.

### Objektum képek

Itt olyan képek találhatóak meg, melyeket a játékos maga tud eszközölni, például az *Inventory* és *CharacterView*, majd a *SkillBar*.



A játékos *Inventory*-ja a játék kezdetében. Felső részben látható a karakter, ahogy éppen kinéz, és milyen eszközök találhatóak rajta. [Fentről lefele] látható a fejsze, köpeny, kesztyű végül a zsák helye. Az alsó részen meg látható az *Inventory*, amelynek ez a kezdeti kapacitása, és a zsákkal a maradék három szekció is feloldható majd.



Itt már láthatóak a felvett eszközök a karakterre. Egy felvett fejsze, egy köpeny és két zsák látható felvéve. Az is jól látható, hogy a felvett eszközök kihatással vannak a *CharacterView*-ban látható játékosra, illetve a zsák az *Inventory* kinézetére (azaz a kapacitásra).



Itt az *Inventory* használata látható már. Jelenleg a játékos rendelkezik egy fejszével egy zsákkal, illetve felvett már 3 db aminósavat is, mely belekerült az *Inventory*-jába. Ha az *Inventory-*ba például eszköz kerül be, akkor annak a felvételét duplaklikkel lehet megvalósítani, illetve ugyanúgy a levétel is így működik. *Inventory*-ból ki is lehet dobni dolgokat, ha már nincs elegendő hely, hiszen a már kicsorbult fejsze nem fog felhalmozódni a már táskában lévő, de még nem használt fejszére, ezért is kell ilyen sok hely.



Így néz ki a kezdetben az úgynevezett *SkillBar*, azaz, ahol az ágensek készítése, tárolása, és az épp karakterre ható ágensek mutatása a felhasználó felé folyik. Bal oldalsó körbe kerül bele az éppen aktuálisan a karakterre ható ágens, lehet az saját maga által feltett ágens, vagy mások által rátett, és lent a bal alsó sarokban jelenik majd meg, hogy hány másodperc maradt hátra még az adott ágens hatóidejéből. Majd [balról jobbra] a belső sávban látható a Vitustánc ikonja, a bénulás, felejtés, végül az immunitás. Itt már két kis rubrika is tartozik egy adott ágenshez, ami arra lesz jó majd, hogyha egy adott ágensből több is elkészült, akkor felső rubrika fogja számlálni a darabszámot, hogy jelenleg hány darab van a játékosnál, alsó rubrikába, meg elkezd visszaszámolni, hogy mennyi ideig lehet még az adott ágenst felhasználni. Mindig a legrégebben elkészült ágens került a „legtetejére a listának”. így ha az elhasználódik, akkor az alatt levő visszaszámlálója váltja fel.



Itt már egy aktív, játék közepén elképzelhető *SkillBar* látható különböző jelzésekkel. Látható, hogy jelenleg a virológusra 1 körig még hat a bénulás, illetve rendelkezik 1 darab Vitustánc támadással, amelyet még 2 körig felhasználhat, 2 darab felejtéssel, melyek közül a legrégebben elkészültet még 3 körig használhatja fel, illetve látszik még, hogy immunitásból szerzett egy genetikai kódot, viszont még nem készítette el az ágenst, melyet a jobb oldalt látható egy gomb, amin egy Erlenymeyer üvegcse van, ha arra rákattintunk, akkor a megpróbál egy ágenst legenerálni, azaz megnézi, hogy van-e megfelelő genetikai kód, illetve megfelelő mennyiségű anyag, és ha van, akkor elkészül az ágens, és hozzá a körös visszaszámláló. Ekkor felkerül annál a gombnál egy 1-es a bal felső sarokban található rubrikába.

### Háttérképek

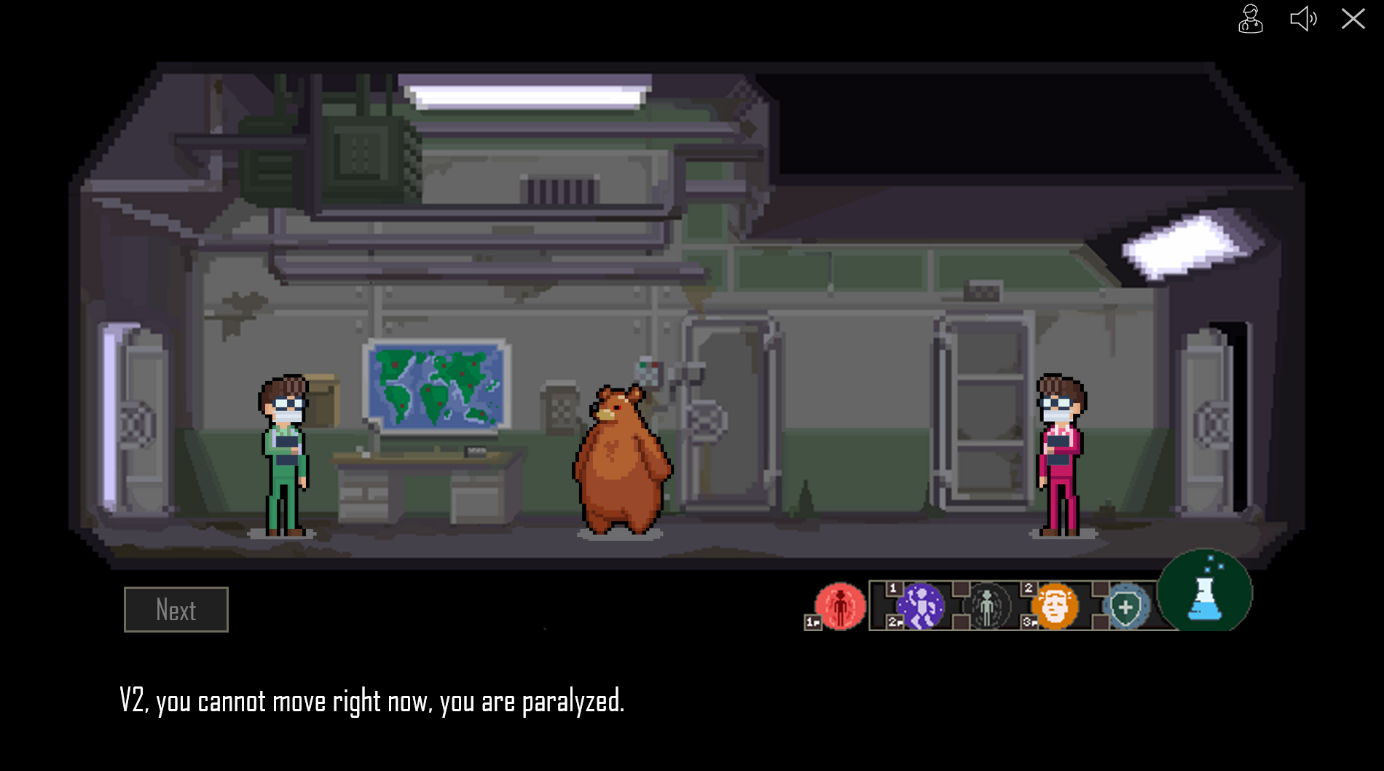
Itt találhatóak a mezők maguk, ugyanis nem „tile game” szerű játékot álmodtunk meg, hanem kihoztuk az egész játékot egy „click and point adventure game”-ként a szebb megjelenítés és a jobb játékélmény érdekében, továbbá nekünk is egyszerűbb volt így elkészíteni, mintsem tile. Első körben itt most csak egy mezőtípust tettünk be példának (sima default/üres mező), majd ugyanazon a mező egy, majd több játékossal (medvetánccal fertőzöttel is), *SkillBar*ral, *Inventory* plusz *CharacterView* meghívásával.



Itt látható az üres háttérkép, amely a háttér lesz az üres mezők során. Még egy kis javítás, további árnyékolások eszközölve lesznek a rendes játékba helyezés előtt.



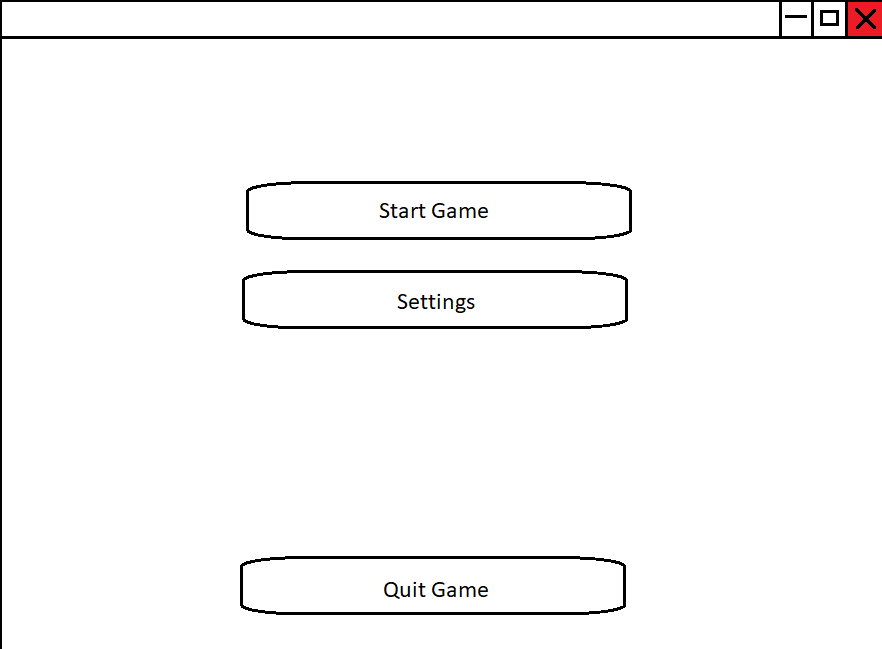
Játék kezdetében nagyjából ilyen látvány fog tárulni a játékosok elé. Lesz egy üres *SkillBar*, egy üres karakter, tehát nem lesz még rajta semmilyen felszerelés, illetve egy üres *Inventory* is. Ha végzett az adott köri feladatával a *Next*, gombbal tud majd a következő játékosnak átadni a stafétát. Ha az ajtókra kattint, azzal tud majd helyszínt váltani, természetesen az, hogy melyik ajtó mögött mi van, azt nem tudhatja előre. A fényesen fehéren világító ajtó lesz mindig az, ahonnan érkezett. Fent a jobb felső sarokban a kis orvos ikonnal tudja előhívni a karakter-tároló panelt, ami látható a képen is jelenleg, illetve a hang gombbal tudja elnémítani vagy éppen hagyni, hogy a játékban bent lévő zene fusson vagy sem. Az x gomb segítségével tud kilépni a játékból, viszont, ha eközben még van bent más játékos, akkor nekik a játék folytatódik tovább.



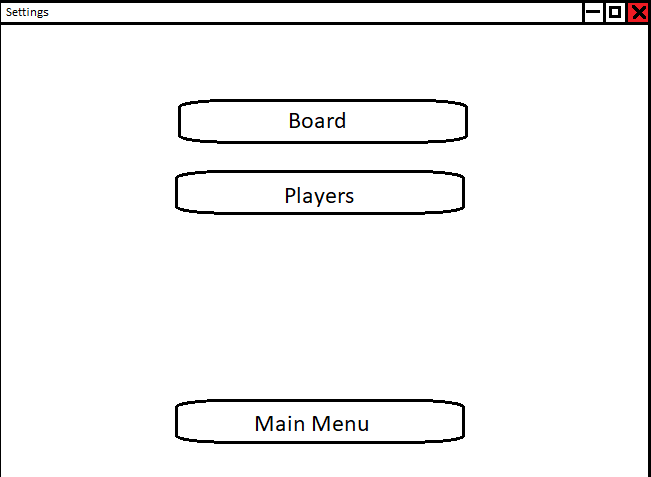
Itt a képen már egy játék közbeni verziót láthatunk. Jelenleg a karakterünk le van bénulva, és a lenti szöveges szekció is szól ezzel kapcsolatban. Ilyenkor nem tudunk rányomni a *Next* gombra, mert automatikusan tovább lépteti a következő játékosra. A képen még látható, hogy van a pályán jelenleg egy medvevírussal fertőzött játékos is, illetve szintén egy, még medvevírussal nem megfertőződött játékos.

### Menü

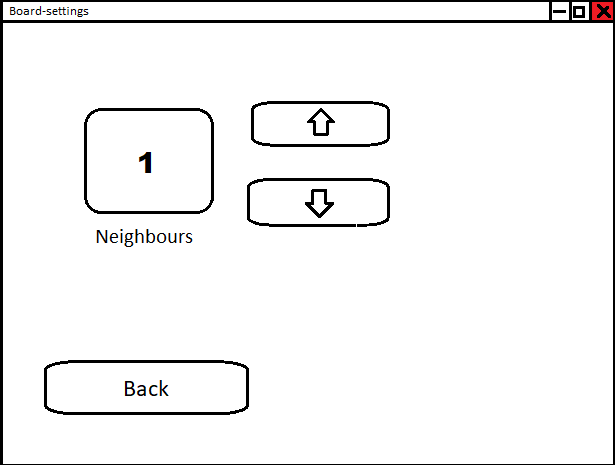
Itt látható egy skicc terv a játék betöltését követő menü rendszerről, és almenüikről.



A játék betöltését követően ez a három menüpont vár majd minket, amelyek viszonylag egyértelműen jelzik, hogy mi történik a kattintásukat követően. A *Start Game*-et kiválasztva elkezdődik a játék, és átvált a fentebb látható nézetre.

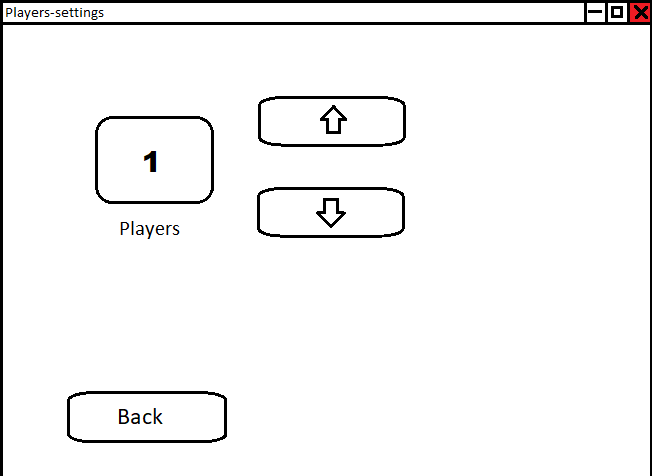


Abban az esetben, ha a *Settings*re kattintunk, átmegy egy másik menübe, ahol választhatunk aközül, hogy a pálya kinézetén változtatunk, a játékosok számán, vagy visszamegyünk a főmenübe. Főmenü esetében az előző képet láthatjuk majd.



**4**

Ha a *Board*ra megyünk, akkor megjelenik ez a nézet, ahol beállíthatjuk, hogy egy mezőnek hány szomszédja lehessen (magyarán milyen alakzatú legyen maga a mező). A default érték 4, a max pedig 6, ezek között nyilakkal lehet mozogni, a bevitt értéket, pedig automatikusan eltárolja. Ha elérte a minimumot, vagy maximumot, akkor a gomb szürkére vált, és nemkattinthatóvá válik. A *Back* gombbal az előző menübe térünk vissza.

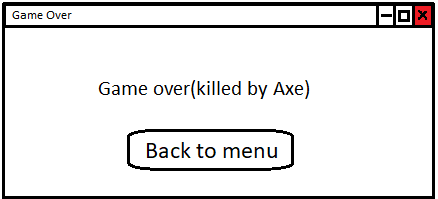


**2**

Ha a *Players*re megyünk, akkor megjelenik a már előbbinél látott nézet. Itt meg lehet adni, hogy maximum hány játékos csatlakozhasson be a játékba és ezek alapján fogja a játék indítását követően legenerálni őket. A default érték 2 (tehát legalább egy ellenfél lehessen), a max pedig 4, ezek között nyilakkal lehet mozogni, a bevitt értéket, pedig automatikusan eltárolja. A *Back* gombbal az előző menübe térünk vissza.

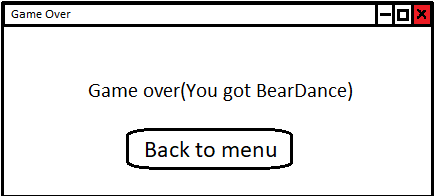
### Játék vége

A játék négy különböző kimenetellel végződhet.



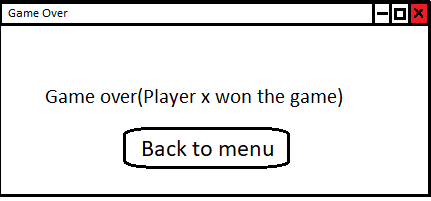
OK

Első eset, amikor a karaktert megölik egy fejszével és úgy ér véget a játék neki. Azonban itt az OK gomb lenyomását követően a többieknek még folytatódik a játék, így automatikusan a soron lévő következő játékost tölti be a játék.

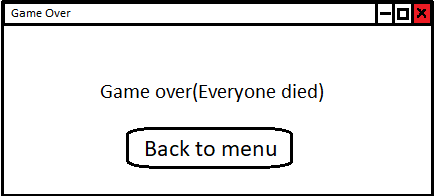


OK

A második eset, hogy medvetánccal fertőződik meg, ebben az esetben az adott játékos már többet nem tud játszani, de az előzővel ellentétben, itt a játékos karaktere még játékban marad, „zombi játékos” lesz belőle.



A harmadik, hogy egy játékos sikeresen összegyűjti az összes genetikai kódot. Ebben az esetben kiíródik a képernyőre, hogy x megnyerte a játékot, így a játék mindenki számára véget ér. Ilyenkor ténylegesen vége a játéknak, és az *OK* gombot felváltja a *Back to menu*, hiszen már nincs mivel játszani.



Ez abban az esetben lép fel, ha mindenki meghal, vagy az utolsó játékos is medvetánccal fertőzött lesz. Itt is hasonlóan az előző verzióban már nem *OK* gomb van.

## A grafikus rendszer architektúrája

### A felület működési elve

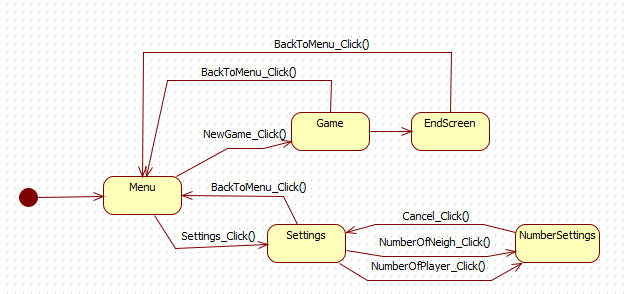
A játék maga egy ablakban fog végig működni a menütől az utolsó percéig a játéknak. Esetlegesen lesznek olyan panelek, amelyek választ, vagy valamilyen fajta reagálást válnak a felhasználótól (pl. mikor megtámadsz valakit, akkor a játék átvált az ő szemszögére, és akkor kap egy ilyen üzenetet, hogy megtámadtak, viszont van nálad egy köpeny, el szeretnéd használni. Ebben az esetben a felhasználónak az igen/nem válaszgombok között kell majd visszareagálnia. Illetve, ha meghívjuk a karakternézőt, akkor az egy előreugró ablakként fog működni, így azt be kell zárni a rajta levő x gomb segítségével). Az egész játékot két módon is bele lehet zárni. A menüben található *Quit Game* segítségével is, illetve az egész játék ablakán található x-gombbal. A játék maga meg van rajzolva, ahogy a háttér is, viszont átlátszó gombokat fogunk helyezni rájuk, amelyek segítik majd a játékost az interakciók tényleges végrehajtásában, így technikailag javarészt mindenhova átlátszó gombokat fogunk elhelyezni a játék szebb megjelenítése érdekében.

### A felület osztály struktúrája

## A grafikus objektumok felsorolása

### MainWindow

* **Felelősség(ek)**
* A program fő ablakának megjelenítését és a játék lefolyásának megjelenítését és vezérlését megvalósító ablak. Ebben valósul meg a menünek, a játéknak, a beállításoknak és a játék befejezését mutató képernyő kirajzolása. Ezek közötti átmenetek átlátását segíti az alábbi state-chart diagram:



* **Ősosztály(ok)**
* JFrame
* **Interfész(ek)**
* Observer
* **Attribútum(ok)**
* - selectedImage:
* - selectedObject:
* TODO HASHTÁBLÁK VALAHOLI EMLÍTÉSE
* - numberOfNeighbors: int : A játékban létrejövő mezők szomszédainak a számát tároló int. Alapértelmezett esetben az értéke 4. Minimális értéke 3, maximális értéke 6.
* - numberOfPlayers: int : A játékot játszó játékosok száma. Minimális értéke 2, maximális értéke 4. Alapértelmezett értéke 2.
* - neighborSet: boolean: A játék beállításához egy segítő boolean. True esetén a szomszédok számát, false esetén a játékosok számát lehet beállítani.
* - windowState: Az ablak állapotát tároló enumeráció. Lehetséges értékei:
  + Menu
  + Game
  + Settings
  + NumberSettings
  + EndScreen
* - musicPlay: boolean: A játék zenéjének lejátszásához segítő boolean. True esetén zene hallatszik, míg false esetén a zene nem szól.
* - mainPanel: JPanel: A MainWindow főnézetét megvalósító panel. Ezen látszódik a menü, a beállítások, a és a játékot záró képernyő. Külön vezérlőket nem tartalmaz.
* - textPanel: JPanel: Ezen a panelen olvashatóak a játék hibaüzenetei a felhasználó számára. A játékon kívül a tartalma üres.
* - skillPanel: JPanel: Itt látható a SkillBar által megjelenített képet. A játékon kívül a tartalma üres
* - skillBar: SkillBar: A MainWindow által eltárolt SkillBar.
* **Metódus(ok)**
* **void StartGame\_Click():** Létrehoz egy új játékot a beállításokban szereplő adatokkal, és átállítja a windowState értékét Game-re. Ezek után újrarajzoltatja az ablakot. Láthatóság: +
* **void Settings\_Click():** Átállítja a windowState értékét Settings-re, majd újrarajzoltatja az ablakot. Láthatóság: +
* **void Exit\_Click():** Kiléptet a programból. Láthatóság: +
* **void** **NumberOfNeigh\_Click():** Átállítja a windowState értékét NumberSettings-re, átállítja a neighborSet értékét true értékre, majd újrarajzoltatja az ablakot. Láthatóság: +
* **void NumberOfPlayer\_Click():** Átállítja a windowState értékét NumberSettings-re, átállítja a neighborSet értékét false értékre, majd újrarajzoltatja az ablakot. Láthatóság: +
* **void BackToMenu\_Click():** Átállítja a windowState értékét Menu-re, majd újrarajzoltatja az ablakot. Láthatóság: +
* **void UpArrow\_Click():** A neighborSet értéke alapján a numberOfNeighbors vagy a numberOfPlayers értékét növeli, ha az még lehetséges. Ha nem, akkor változatlanul hagyja. Láthatóság: +
* **void DownArrow\_Click():** A neighborSet értéke alapján a numberOfNeighbors vagy a numberOfPlayers értékét csökkenti, ha az még lehetséges. Ha nem, akkor változatlanul hagyja. Láthatóság: +
* **void Cancel\_Click():** A windowState értékét Settings-re állítja, majd újrarajzoltatja az ablakot. Láthatóság: +
* **void Click():** Mivel az ablaknak különböző megjelenései lehetnek, ezért a windowState különböző állapotaiban különböző viselkedése van. Láthatóság: +
  + Menu: Megvizsgálja a kattintás koordinátáit. Ha egy gomb helyére esik, akkor meghívja a hozzá tartozó Click metódust. (Start game, Settings, PlayMusic, Exit). Ha ezeken kívüli pozíció, akkor nem csinál semmit.
  + Game: TODO
  + Settings: Megvizsgálja a kattintás koordinátáit. Ha egy gomb helyére esik, akkor meghívja a hozzá tartozó Click metódust. (NumberOfNeigh, NumberOfPlayer, PlayMusic, BackToMenu). Ha ezeken kívüli a pozíció, akkor nem csinál semmit.
  + NumberSettings: Megvizsgálja a kattintás koordinátáit. Ha egy gomb helyére esik, akkor meghívja a hozzá tartozó Click metódust (UpArrow, DownArrow, PlayMusic, Cancel). Ha ezeken kívüli a pozíció, akkor nem csinál semmit.
  + EndScreen: Megvizsgálja, hogy a kattintás koordinátája a gombra esik-e. Ha igen, meghívja a BackToMenu\_Click() metódust, ellenkező esetben nem csinál semmit.
* **void PlayMusic\_Click():** Negálja a musicPlay értékét. Ha az így kapott érték false, akkor leállítja a zenét, ha true, akkor elindítja azt. Láthatóság: +
* **void repaint():** Újra rajzoltatja a paneleket. Láthatóság: -
* **void mainPanelPaint(Graphics):** A mainPanel kirajzoltatásáért felelős. A játék jelenlegi állásától függően választja ki a megfelelő képet a kirajzoltatáshoz. Láthatóság: -
* **void skillPanelPaint(Graphics):** A meghívja a skillBar Update metódusát, majd a getImage által kapott képet kirajzoltatja. Láthatóság: -

### SkillBar

* **Felelősség(ek)**
* A virológus által létrehozható ágenseknek a vizualizációját segítő osztály. Segítségével jelenik meg a grafikus interfészben leírt SkillBar. A játékban éppen aktív virológustól lekérdezett adatai alapján változik a működése.

(Ha ismeri az adott ágenshez tartozó genetikai kódot, akkor az annak megfelelő ikon látható, ellenkező esetben annak “kiszürkített” állapota. A createAgent-hez tartozó ikon az addig szürke, amíg a virológus egyetlen genetikai kódot sem ismer.)

* **Ősosztály(ok)**
* **Interfész(ek)**
* **Attribútum(ok)**
* - selected: String: Az éppen aktuálisan kiválasztott ágens típusa.
* - createClicked: boolean
* **Metódus(ok)**
* **void Immunity\_Click():** Megvizsgálja, hogy az éppen aktív virológus ismeri-e az Immunity genetikai kódját. Ha igen, akkor, ha a createClicked értéke true, akkor megpróbálja létrehozni a az adott ágenst, aminek sikertelenségéről a MainWindow alsó részéről értesülhet a felhasználó. Ezen kívül a selected értékét null-ra változtatja. Ha a createClicked értéke false, akkor a selected értéke “Immunity”. Láthatóság: +
* **void Paralysis\_Click():** Megvizsgálja, hogy az éppen aktív virológus ismeri-e az Paralysis genetikai kódját. Ha igen, akkor, ha a createClicked értéke true, akkor megpróbálja létrehozni a az adott ágenst, aminek sikertelenségéről a MainWindow alsó részéről értesülhet a felhasználó. Ezen kívül a selected értékét null-ra változtatja. Ha a createClicked értéke false, akkor a selected értéke “Paralysis”. Láthatóság: +
* **void VitusDance\_Click():** Megvizsgálja, hogy az éppen aktív virológus ismeri-e az VitusDance genetikai kódját. Ha igen, akkor, ha a createClicked értéke true, akkor megpróbálja létrehozni a az adott ágenst, aminek sikertelenségéről a MainWindow alsó részéről értesülhet a felhasználó. Ezen kívül a selected értékét null-ra változtatja. Ha a createClicked értéke false, akkor a selected értéke “VitusDance”. Láthatóság: +
* **void Create\_Click():** Ha a selected értéke null, akkor a createClicked értékét true-ra állítja. Ha a selected értéke “Immunity”, “Paralysis” vagy “VitusDance”, akkor megpróbálja létrehozni a kiválasztott ágenst és a selected értékét null-ra állítja. Ha a selected értéke valahogyan értelmezhetetlen adatot tárol, akkor annak értékét null-ra állítja és tájékoztatja a felhaszálót a MainWindow alsó részén. Láthatóság: +
* **BufferedImage getImage():** A virológus jelenlegi ismert genetikai kódjai alapján visszaadja a MainWindownak a megfelelő képet a kirajzoláshoz. Láthatóság: +
* **void Update():** A SkillBar újra lekérdezi az aktuális virológust, és annak az ismert és eltárolt ágenseit. Láthatóság: +

### Skill

* **Felelősség(ek)**
* A virológusok által létrehozott ágenseknek a vizualizációját segítő osztály. Segítségével jelenik meg a grafikus interfészben leírt SkillBar bal oldalán az összes ágens, amelyek a játékban éppen aktív virológus számára rendelkezésre állnak.
* **Ősosztály(ok)**
* **Interfész(ek)**
* **Attribútum(ok)**
* -
* **Metódus(ok)**
* **void Immunity\_Click():**

### WindowDraw

* **Felelősség(ek)**
* Ez az enumeráció megadja, hogy éppen melyik ablak aktív a lehetséges ablakok közül: menü (Menu), játék (Game), beállítások (Settings), virológusok / szomszédok számának beállítása (NumberSettings) vagy játék vége (EndScreen).

### Skill

* **Felelősség(ek)**
* **Ősosztály(ok)**
* **Interfész(ek)**
* **Attribútum(ok)**
* **Metódus(ok)**
* **void Immunity\_Click():**
* **Felelősség(ek)**
* **Ősosztály(ok)**
* **Interfész(ek)**
* **Attribútum(ok)**

**Metódus(ok)**

**void Immunity\_Click():** Megvizsgálja, hogy az éppen aktív virológus ismeri-e az Immunity genetikai kódját. Ha igen, akkor a

## Kapcsolat az alkalmazói rendszerrel

### MainWindow

### SkillBar

### WindowDraw

### Virologist\_changed

### Timer\_changed

### MyKeyListener

### MyMouseListener

### CIMouseListener

### CIView

## Ütemterv

|  |  |  |  |
| --- | --- | --- | --- |
| **Határidő** | **Feladat** | **Pont** | **Felelős** |
| **febr. 28.** | Követelmény, projekt, funkcionalitás - beadás | 10 | Csia |
| **márc. 7.** | Analízis modell kidolgozása I. - beadás | 20 | Alpek |
| **márc. 16.** | Analízis modell kidolgozása II. - beadás | 30 | Marton |
| **márc. 21.** | Szkeleton tervezése - beadás | 20 | Litavecz |
| **márc. 28.** | Szkeleton elkészítése - beadás | 20 | Csia |
| **ápr. 4.** | Prototípus koncepciója – beadás  Szkeleton bemutatás | 20 | Alpek |
| **ápr. 11.** | Részletes tervek - beadás | 45 | Marton |
| **ápr. 25.** | Prototípus készítése, tesztelése | 35 | Litavecz |
| **máj. 2.** | Grafikus változat tervei – beadás  Prototípus - beadás és a forráskód, a tesztbemenetek és az elvárt kimenetek herculesre való feltöltése | 30 | Ruskó |
| **máj. 16.** | Grafikus változat készítése | 40 |  |
| **máj. 18.** | Egységes dokumentáció - beadás és bemutatás  Grafikus változat - beadás és a forráskód herculesre való feltöltése, és teljes házi bemutatás | 30 |  |

## Napló

|  |  |  |  |
| --- | --- | --- | --- |
| **Kezdet** | **Időtartam** | **Résztvevő(k)** | **Leírás** |

|  |  |  |  |
| --- | --- | --- | --- |
| 2022.04.11 17:40 | 1 óra | Alpek, Csia, Ruskó | Megbeszélés: Grafikus rész alapötletek megvitatása. |
| 2022.04.11 – 05.02 | Kb. 30 óra | Csia | Feladat: Elemek, hátterek, karakterek, objektumok egyedileg megtervezése, Pixelart nevű program segítségével megrajzolásuk. |
| 2022.05.01 11:20 | 1,5 óra | Csia, Ruskó | Megbeszélés: Osztálydiagram alapvető átbeszélése, alapvető ötletek egyeztetése a grafikai kinézettel kapcsolatban. |
| 2022.05.01 13:40 | 2 óra 20 perc | Csia, Ruskó | Megbeszélés: További ötletek, problémák megvitatása a grafikus kinézettel kapcsolatban, osztálydiagram és állapotgépek megrajzolása a megbeszéltek szerint. |
| 2022.05.01 16:00 | 3 óra | Ruskó | Feladat: Osztálydiagram leírások elkészítése. |
| 2022.05.01 17:20 | 1,5 óra | Csia | Feladat: Grafikus interfész szekció összeállítása, szövegezések megírása. |
| 2022.05.02. 17:00 | 6 óra | Alpek, Csia, Marton, Ruskó | Megbeszélés: Osztálydiagram teljes átszervezése, feladatok szétosztása. |
| 2022.05.03 06:00 | 1 óra | Alpek | Feladat: Menü és játék vége skiccek elkészítése. |
| 2022.0503 11:00 | 2 óra | Csia | Feladat: Menü és játék vége részekhez szövegezés elkészítése, A felület működési elv rész megírása. |
| 2022.05.03 15:00 | 8 óra | Marton | Feladat: Osztálydiagrammok leírásainak elkészítése. |
| 2022.05.03 16:00 | 8 óra | Ruskó | Feladat: Szekvenvciadiagrammok megrajzolása |

## Százalékos teljesítés

|  |  |
| --- | --- |
| **Név** | **Százalék** |

|  |  |
| --- | --- |
| Alpek | 20% |
| Csia | 20% |
| Litavecz | 20% |
| Marton | 20% |
| Ruskó | 20% |