**11. Grafikus felület specifikációja**

15 – jgoldfisch

Konzulens:

Bodó Zsófia

Csapattagok

|  |  |  |
| --- | --- | --- |
| Mátyás Gergely | IL21NI | matyasg97@gmail.com |
| Horváth Ákos | DKILK6 | horvath.akos1997@gmail.com |
| Gurubi Barnabás | DXEXVR | gurubibarni@gmail.com |
| Tolnai Márk | ID61MK | tolesz11@windowslive.com |
| Bertalan Bálint | HNN9GA | blintber@gmail.com |

2018. május 1.

# Grafikus felület specifikációja

A grafikus felület textúráinak elkészítéséért köszönet jár Polgár Alex grafikusnak.

## A grafikus interfész

A játék elindítását követően egy menübe jutunk, ahol új játékot lehet kezdeni, vagy ki lehet lépni a programból. Hívjuk ezt a továbbiakban főmenünek.

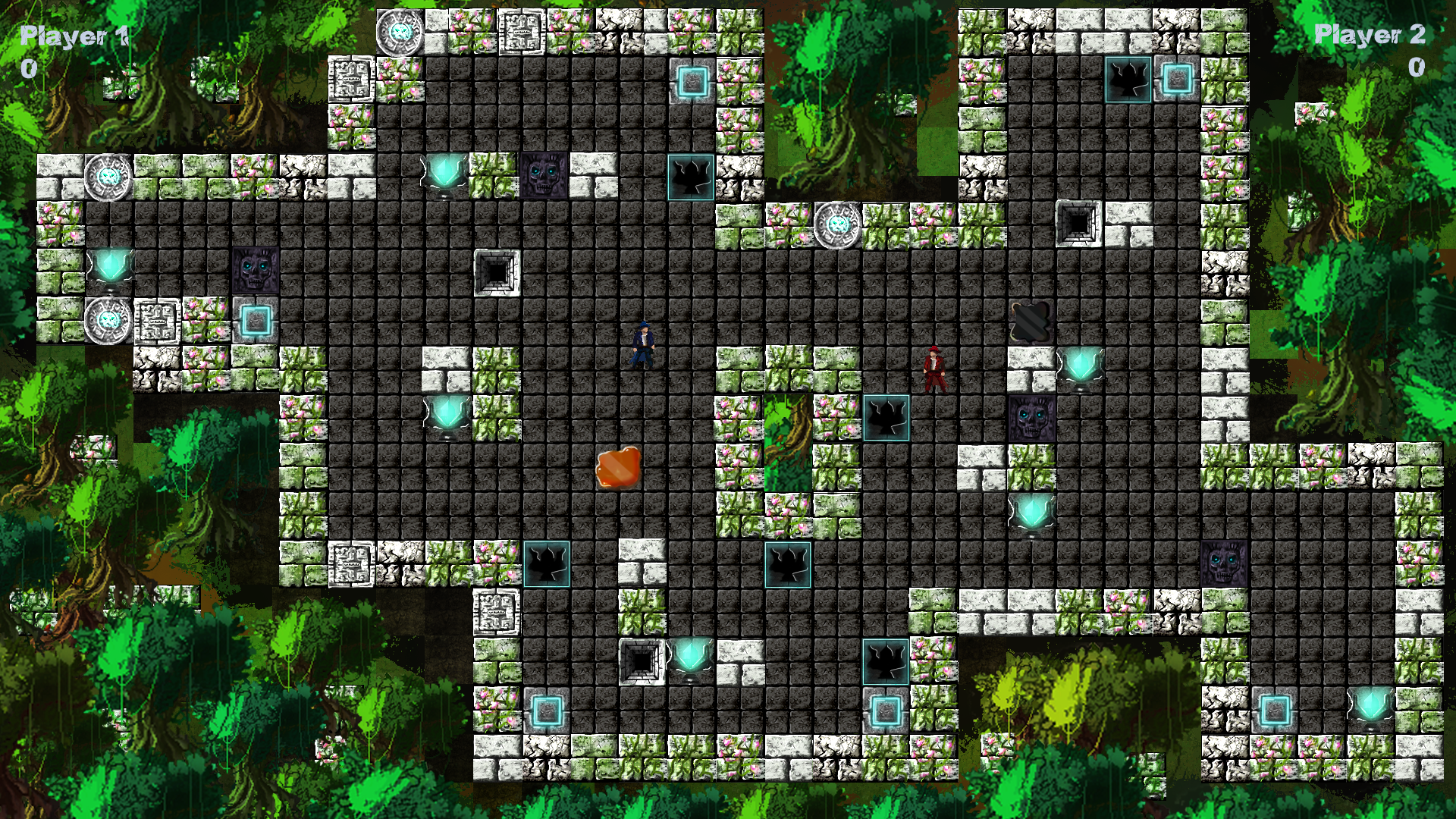
1. ábra: A főmenü

A főmenűben tálálható két gomb a következő: Exit, New Game

Korábbinak jelentése egyértelmű, miután rákattintottunk bezárja a programot. Utóbbi egy második menüre visz minket, melyből kiválaszthatjuk, hogy melyik pályán szeretnénk játszani. (pályaválasztó).

2. ábra A pálya választó menü

A megfelelő gombra kattintva a megfelelő pálya jön be, míg a Back gomb, a főmenübe visz minket vissza. A pályák ugyan azon elven épülnek fel. Betöltés után egy ehhez hasonló pályát láthatunk.



3. ábra: Jungle Ruin pálya, minta elemekkel

Természetesen a valós játékban a pályán alapvetően egyes elemek nem lesznek láthatóak, mint például a méz és az olaj. De nézzük végig, hogy milyen elemek találhatóak a pályán:

|  |  |
| --- | --- |
|  | A játékosok textúrái. Felül az egyes játékosé, alul pedig a kettes játékosé. |
|  | A dobozok (shardok), melyeket a helyükre kell tolni. |
|  | A dobozok lerakóhelye. |
|  | A sima kapcsoló nélküli lyuk, vagy a lenyomott kapcsolóval rendelkező lyuk |
|  | A zárt lyuk, melyen biztonságosan átmehet egy játékos, vagy doboz. |
|  | A kapcsoló, mely aktiválhatja a lyukat. |
|  | A fenti az olaj, az alsó pedig a méz mezőmódosító, melyek a mező tapadását befolyásolják. |

## A grafikus rendszer architektúrája

[A felület működésének elve, a grafikus rendszer architektúrája (struktúra diagramok). A struktúra diagramokon a prototípus azon és csak azon osztályainak is szerepelnie kell, amelyekhez a grafikus felületet létrehozó osztályok kapcsolódnak.]

### A felület működési elve

A felület alapjául a Model-View-Controller tervezési minta szolgál. Az eddigiek során a játék logikája készült el, ez felel meg a **modell-**nek, ez írja le a játék működését, számontartja az állapotát. A játék belső állapota, működése érezhetően nem függ a kinézettől (hiszen eddig karakteresen vezéreltük GUI nélkül), emiatt a **view**-nak minél jobban el kell különülnie. A játék kívülről való vezérlését a **controller** valósítja meg, ezesetben a billentyűzet kezelése, továbbá a modell és view értesítése a fő feladat.

Alapvetően **push** alapú a működés, vagyis változás esetén (pl. billentyű lenyomás) a controller értesíti a résztvevőket a változásról, a modell a változásnak megfelelően frissíti az állapotát és az új állapot alapján a view is frissül.

A játék megjelenése két nagy részre tagolható, a menüre és a tényleges játéknézetre. A menü egyszerűsége miatt nem valósít meg saját MVC mintát, hiszen kevés funkciója nem kívánja meg e három rész elkülönülését.

A játéknézet rész view-ja, vagyis nézete a modellben (játéklogika) megtalálható osztályok, entitások grafikus megfelelőiből (reprezentációiból) fog állni. Azzal, hogy a view-ban direktben nem jelennek meg a játéklogika elemei, nem határozzák meg a kinézetet, így az könnyebben módosítható, esetleg teljes egészében cserélhető.

Magához a **Game**-hez tartozik a **View**, ami tárolja a rajzolható **Drawable** absztrakt ősosztályból származó objektumokat (ezekhez tartoznak a modellben szereplő business objektumok). A program főablakját a **MainFrame** osztály reprezentálja, ebben található a **Menu** (a menü egészét megvalósító osztály) és a **Game**. A külső vezérlést megvalósító **Controller** szintén a **Game** része.

### A felület osztály-struktúrája

[Osztálydiagram. Minden új osztály, és azon régiek, akik az újakhoz közvetlenül kapcsolódnak.]

## A grafikus objektumok felsorolása

[Az új osztályok felsorolása. Az régi osztályok közül azoknak a felsorolása, ahol változás volt. Ezek esetén csak a változásokat kell leírni.]

### Osztály1

#### Felelősség

[Mi az osztály felelőssége. Kb 1 bekezdés. Ha szükséges, akkor state-chart is.]

#### Ősosztályok

[Mely osztályokból származik (öröklési hierarchia)]

Legősebb osztály → Ősosztály2 → Ősosztály3...

#### Interfészek

[Mely interfészeket valósítja meg.]

#### Attribútumok

[Milyen attribútumai vannak]

* **attribútum1**: attribútum jellemzése: mire való, láthatósága (UML jelöléssel), típusa
* **attribútum2**: attribútum jellemzése: mire való, láthatósága (UML jelöléssel), típusa

#### Metódusok

[Milyen publikus, protected és privát metódusokkal rendelkezik. Metódusonként precíz leírás, ha szükséges, activity diagram is a metódusban megvalósítandó algoritmusról.]

* **int foo(Osztály3 o1, Osztály4 o2)**: metódus leírása, láthatósága (UML jelöléssel)
* **int bar(Osztály5 o1)**: metódus leírása, láthatósága (UML jelöléssel)

### Controller

#### Felelősség

Az MVC mintában a controller-t megvalósító osztály. Az egyes inputok alapján végzi az egész működés irányítását. A bemenetek a billentyűzet, illetve a különböző képeken (panel-ek) lévő gombok használatából keletkeznek.

#### Ősosztályok

-

#### Attribútumok

* **-MainFrame frame:** A főablak, ez tartalmazza a legfőbb JPanelt.
* **-Game game:** A játéklogika legfelsőbb eleme, itt csatlakozik a modell.
* **-KeyBoardHandler:** A billentyűzet eseményeit kezelő osztály, ez értesíti a controllert.

#### Metódusok

* **+ void workerControlled(int workerID, Direction d):** a játékos mozgatását kezdeményező függvény, meghívja a modellben a felelős függvényt.
* **+ void oilDropped(int workerID):** speciális elem (olaj) lerakását indító függvény, meghívja a modellben ezt a tevékenységet végző függvényt.
* **+ void honeyDropped(int workerID):** speciális elem (méz) lerakását indító függvény, meghívja a modellben ezt a tevékenységet végző függvényt.
* **+ void updateView():** a view-t frissítő függvény, hatására az egész view újrarajzolódik.
* **+ void startGame(String mapFile):** új játék kezdeményezését indító függvény, inicializálja a játékot (**Game**), továbbá betölti a textúrákat, kirajzolja a játékképet (**GamePanel**) és megjeleníti azt.
* **+ void endGame():** játék vége esetén hívódik, ekkor megjeleníti a záróképet (**EndPanel**).

### Drawable

#### Felelősség

A kirajzolandó osztályok absztrakt ősosztálya, ismeri a textúráját.

#### Ősosztályok

-

#### Attribútumok

* **Image texture:** az objektumhoz tartozó textúra (kép).
* **JPanel imagePanel:** a kép elhelyezéséért felelős JPanel.

#### Metódusok

* **void draw():** a kép kirajzolását végző függvény.

### G\_Box

#### Felelősség

Egy Box objektum kirajzolása.

#### Ősosztályok

Drawable

#### Attribútumok

* **Image texture:** az objektumhoz tartozó textúra (kép).
* **JPanel imagePanel:** a kép elhelyezéséért felelős JPanel.
* **Box gameObj:** a pályán lévő az az objektum, amihez a kép tartozik.

#### Metódusok

* **void draw():** a kép kirajzolását végző függvény.

### G\_EndField

#### Felelősség

Egy célmező (EndField) kirajzolása.

#### Ősosztályok

Drawable

#### Attribútumok

* **Image texture:** a mezőhöz tartozó textúra (kép).
* **JPanel imagePanel:** a kép elhelyezéséért felelős JPanel.
* **EndField gameObj:** a pályán lévő az a mező, amihez a kép tartozik.

#### Metódusok

* **void draw():** a kép kirajzolását végző függvény.

### G\_HoleField

#### Felelősség

Egy lyuk (HoleField) mező kirajzolása.

#### Ősosztályok

Drawable

#### Attribútumok

* **Image texture:** a mezőhöz tartozó textúra (kép) (a lyuk nyitott állapotban).
* **Image secTexture:** a mezőhöz tartozó textúra (kép) (a lyuk csukott állapotban).
* **JPanel imagePanel:** a kép elhelyezéséért felelős JPanel.
* **HoleField gameObj:** a pályán lévő az a mező, amihez a kép tartozik.

#### Metódusok

* **void draw():** a kép kirajzolását végző függvény.

### G\_Honey

#### Felelősség

Egy Honey objektum kirajzolása.

#### Ősosztályok

Drawable

#### Attribútumok

* **Image texture:** az objektumhoz tartozó textúra (kép).
* **JPanel imagePanel:** a kép elhelyezéséért felelős JPanel.
* **Honey gameObj:** a pályán lévő az az objektum, amihez a kép tartozik.

#### Metódusok

* **void draw():** a kép kirajzolását végző függvény.

### G\_Oil

#### Felelősség

Egy Oil objektum kirajzolása.

#### Ősosztályok

Drawable

#### Attribútumok

* **Image texture:** az objektumhoz tartozó textúra (kép).
* **JPanel imagePanel:** a kép elhelyezéséért felelős JPanel.
* **Oil gameObj:** a pályán lévő az az objektum, amihez a kép tartozik.

#### Metódusok

* **void draw():** a kép kirajzolását végző függvény.

### G\_SimpleField

#### Felelősség

Egy egyszerű mező (SimpleField) kirajzolása.

#### Ősosztályok

Drawable

#### Attribútumok

* **Image texture:** a mezőhöz tartozó textúra (kép).
* **JPanel imagePanel:** a kép elhelyezéséért felelős JPanel.
* **SimpleField gameObj:** a pályán lévő az a mező, amihez a kép tartozik.

#### Metódusok

* **void draw():** a kép kirajzolását végző függvény.

### G\_SwitchField

#### Felelősség

Egy kapcsoló (SwitchField) mező kirajzolása.

#### Ősosztályok

Drawable

#### Attribútumok

* **Image texture:** a mezőhöz tartozó textúra (kép).
* **JPanel imagePanel:** a kép elhelyezéséért felelős JPanel.
* **SwitchField gameObj:** a pályán lévő az a mező, amihez a kép tartozik.

#### Metódusok

* **void draw():** a kép kirajzolását végző függvény.

### G\_WallField

#### Felelősség

Egy fal (WallField) mező kirajzolása.

#### Ősosztályok

Drawable

#### Attribútumok

* **Image texture:** a mezőhöz tartozó textúra (kép).
* **JPanel imagePanel:** a kép elhelyezéséért felelős JPanel.
* **WallField gameObj:** a pályán lévő az a mező, amihez a kép tartozik.

#### Metódusok

* **void draw():** a kép kirajzolását végző függvény.

### G\_Worker

#### Felelősség

Egy munkás (Worker) objektum kirajzolása.

#### Ősosztályok

Drawable

#### Attribútumok

* **Image texture:** az objektumhoz tartozó textúra (kép).
* **JPanel imagePanel:** a kép elhelyezéséért felelős JPanel.
* **Worker gameObj:** a pályán lévő az az objektum, amihez a kép tartozik.

#### Metódusok

* **void draw():** a kép kirajzolását végző függvény.

### GamePanel

#### Felelősség

Az MVC minta view részének egy eleme, maga a játékmenetet megjelenítő panel. Tartalmazza a modellben szereplő entitások kirajzolandó változatait. Ezeket a modell alapján jeleníti meg. JPanel leszármazott, ezzel könnyítve a megvalósítást. Tárolja a pályához tartozó textúrákat is.

#### Ősosztályok

JPanel, MainPanel

#### Attribútumok

* **Image backgroundImage:** a Panel háttérképe.
* **Font font:** a Panelhez tartozó betűtípus.
* **List<Image> textures:** az adott pályához tartozó különböző texttúrák, így csak egyszer szükséges betölteni őket.
* **List<Drawables> staticDrawables:** a pályához tartozó statikus megjelenítendő elemek (pl. fal).
* **List<Drawables> dynamicDrawables:** a pályához tartozó dinamikus megjelenítendő elemek (pl. munkás).

#### Metódusok

* **+ void drawStaticView():** A statikus elemek kirajzolását végző függvény, egyszer kell hívni, hiszen ezek az elemek nem változnak a játék menete során.
* **+ void drawDynamicView():** A dinamikus elemek kirajzolását végző függvény, ez meghívandó a modell minden változása esetén.
* **+ void loadTextures(String viewStyle):** A pályához tartozó textúrák betöltését elvégző függvény.

### KeyEventHandler

#### Felelősség

A billentyűzet kezelését végző osztály, implementálja a KeyListener interfészt.

#### Interfészek

KeyListener

#### Attribútumok

-

#### Metódusok

* **void keyPressed(KeyEvent e):** billentyűzet leütését kezelő függvény, a különböző billentyűknek megfelelő utasítás függvényét hívja a **Controllerben**.

## Kapcsolat az alkalmazói rendszerrel

[Szekvencia-diagramokon ábrázolni kell a grafikus rendszer működését. Konzisztens kell legyen az előző alfejezetekkel. Minden metódus, ami ott szerepel, fel kell tűnjön valamelyik szekvenciában. Minden metódusnak, ami szekvenciában szerepel, szereplnie kell a valamelyik osztálydiagramon.]

## Napló

|  |  |  |  |
| --- | --- | --- | --- |
| **Kezdet** | **Időtartam** | **Résztvevők** | **Leírás** |
| 2010.03.21. 18:00 | 2,5 óra | Horváth  Németh  Tóth  Oláh | Értekezlet.  Döntés: Horváth elkészíti az osztálydiagramot, Oláh a use-case leírásokat. |
| 2010.03.23. 23:00 | 5 óra | Németh | Tevékenység: Németh implementálja a tesztelő programokat. |
| … | … | … | … |