Specifikáció

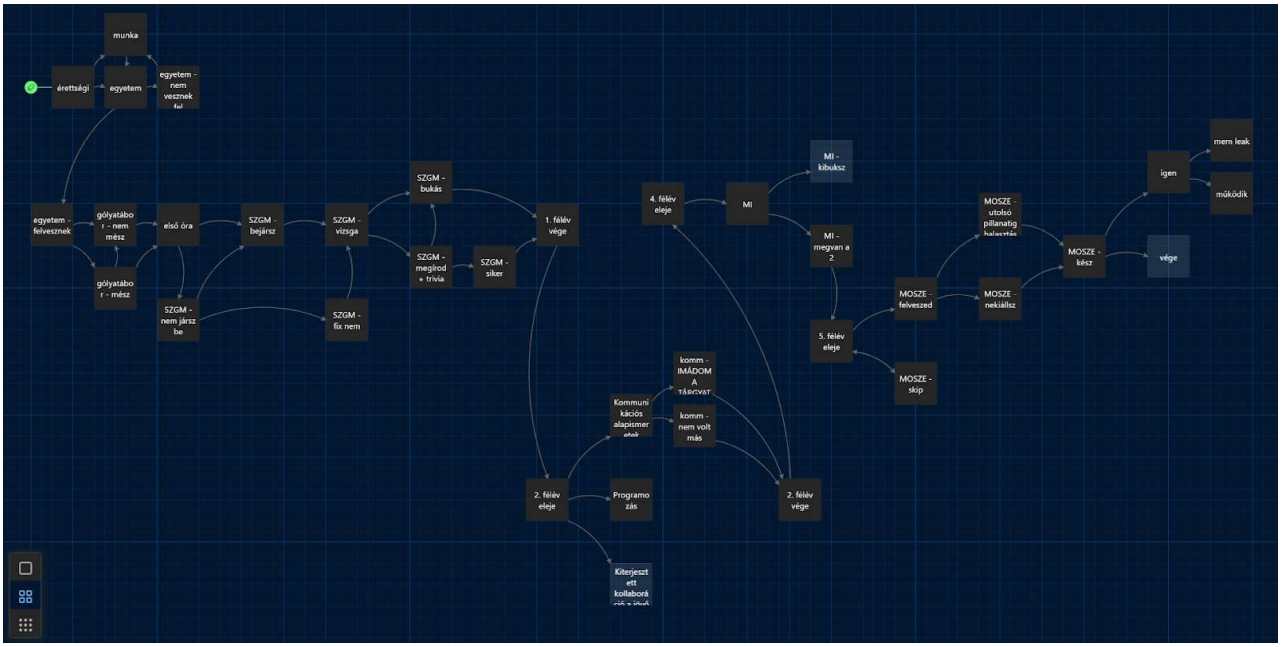
Gitanos

Modern szoftverfejlesztési eszközök, 2022/23 I.

Tartalomjegyzék

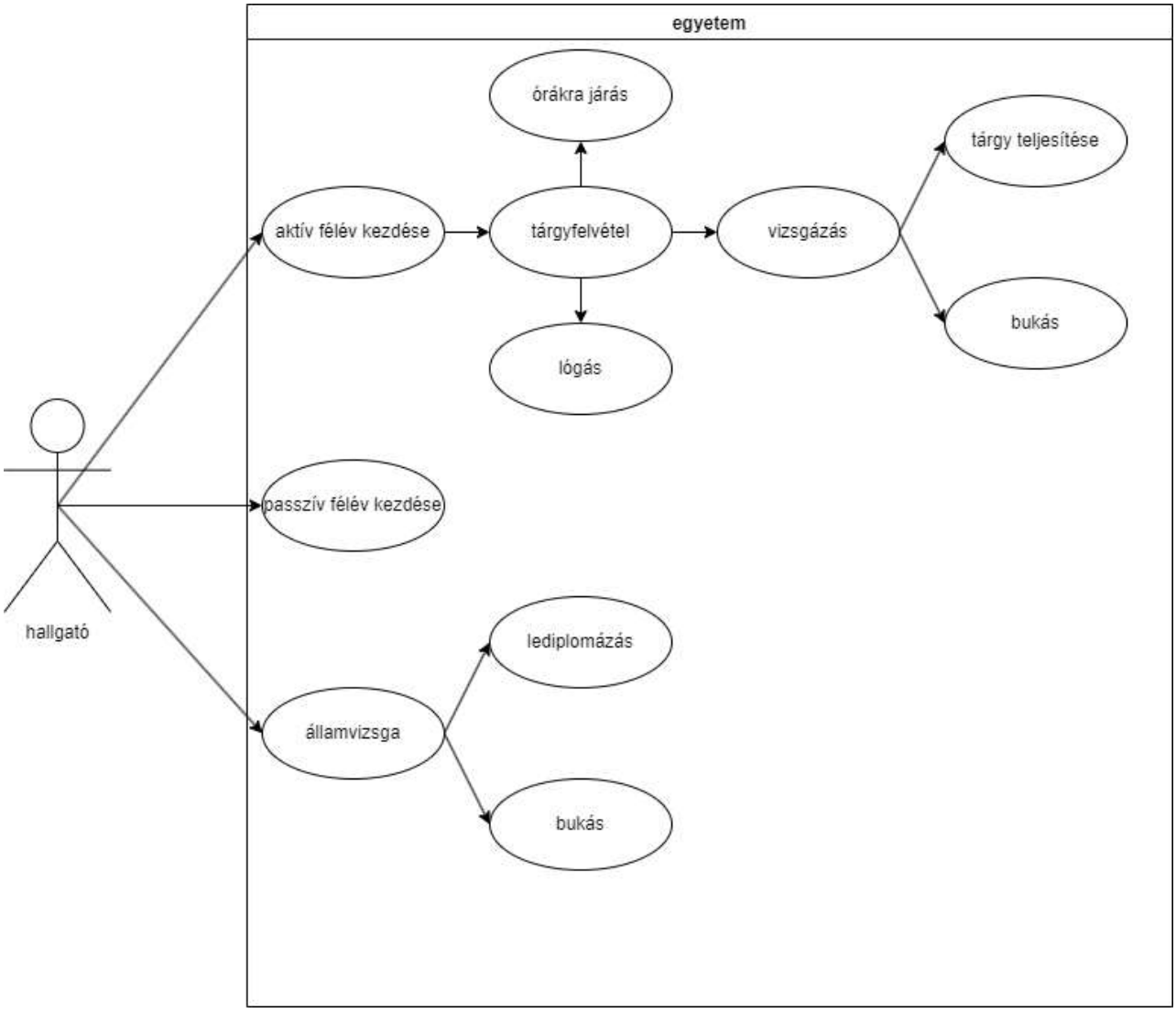
Történet

- A játék azzal kezdődik, hogy feltételezzük hogy a játékos elvégezte az érettségit, majd ezt követően dönt, hogy elmegy e dolgozni, vagy pedig inkább továbbmegy egyetemre tanulni
- Ha felveszik az egyetemre, akkor dönt arról hogy hogy részt vesz e a gólyatáborban vagy sem
- Ezt követően el elérkezünk a játékos egyetemi karrierjének egy kulcs pontjához, ahol



Használati esetek (use case)

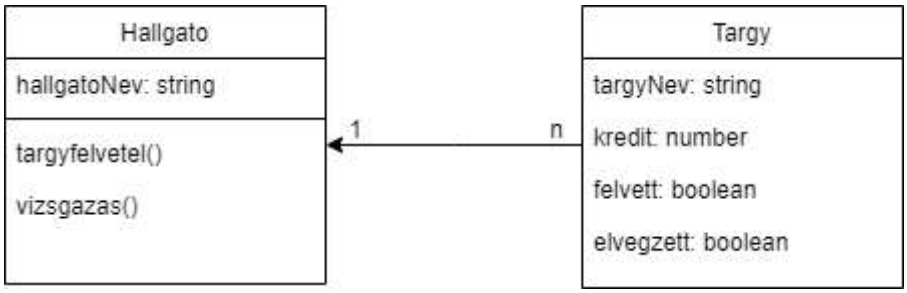
Megnevezés	Prioritás	Leírás
tárgyfelvétel		
órákra járás		növeli adott tárgy elvégzésének esélyét
lógás		csökkenti adott tárgy elvégzésének esélyét
vizsgázás		próbálkozás adott tárgy elvégzésével • kreditszám és előző lépések befolyásolják a sikeresség esélyét • ha sikeres akkor adott tárgy abszolválva • túl sok bukás a játék végét jelentheti
államvizsga		próbálkozás egyetem elvégzésével • előfeltétel a tárgyak teljesítése • ha sikerül, akkor a játék véget ér


egyetem
órákra járás
tárgy teljesítése
aktív félév kezdése
tárgyfelvétel
vizsgázás
bukás
lógás
hallgató
passzív félév kezdése
lediplomázás
államvizsga
bukás

folyt köv :)

Azonosított követelmények

Struktúra (UML class diagram)

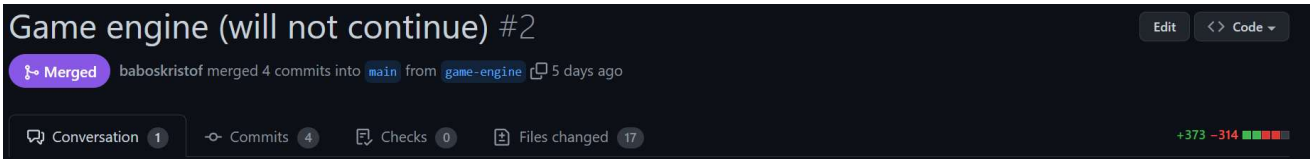


NEM VÉGLEGES

Branch használat

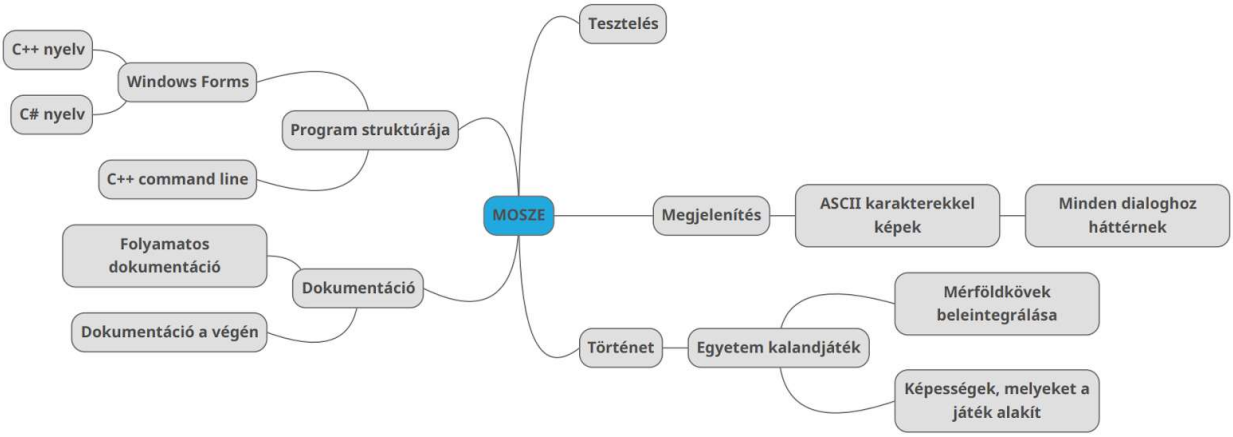
Game Engine V 1.0

Az első koncepció kivitelezéséhez a Windows Forms-al került megvalósításra C++ nyelven. Ez a megjelenítést sokkal elősegítette, viszont nem tudtunk vele dinamikusan fejlődni. Új ötletek ütötték fel a fejüket, ezért ezen ág megszüntetése mellett döntöttünk 3-4 napos munkát átvittünk egy másik projektbe.



Game Engine V 2.0

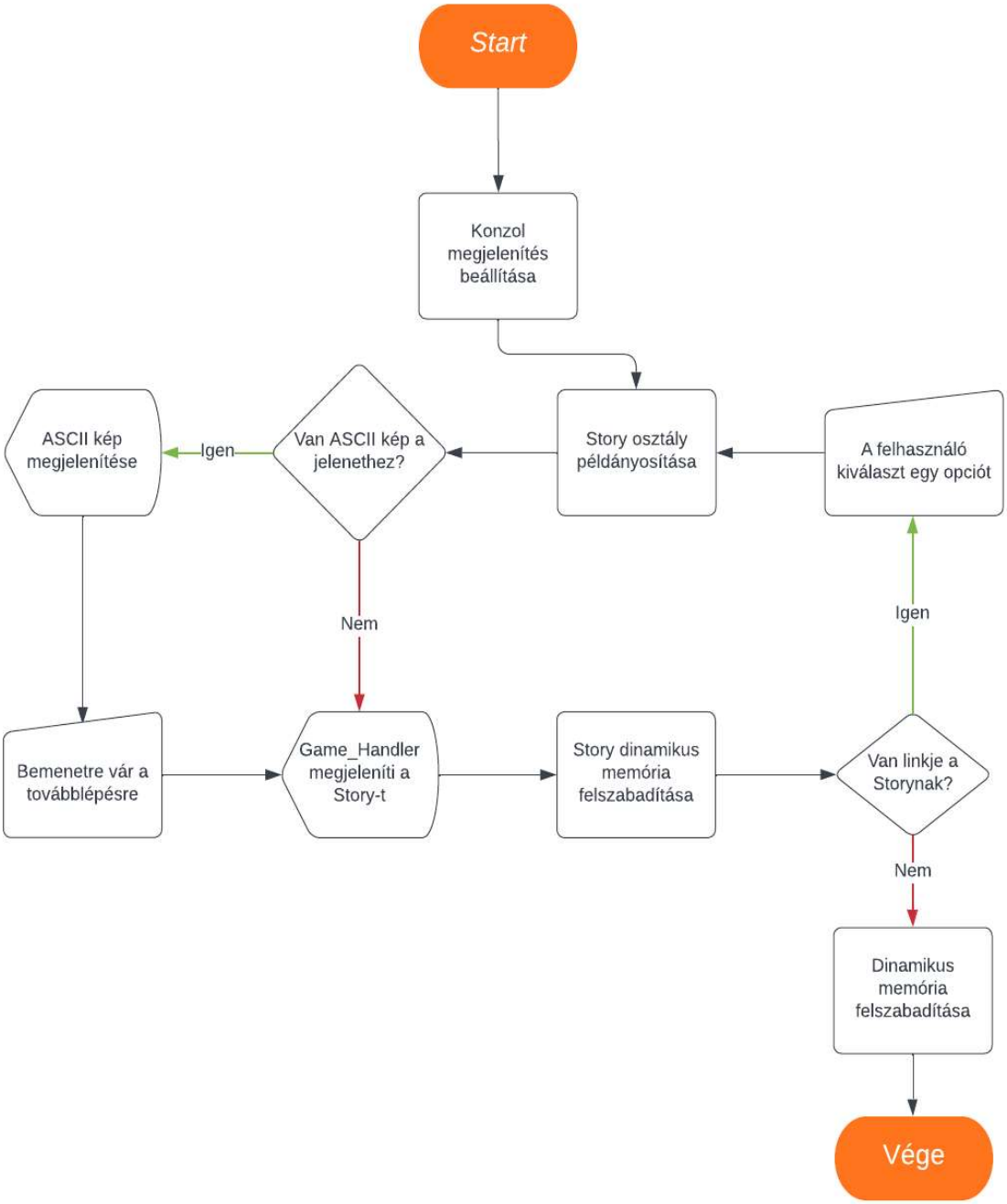
Az első engine bukása után ebben az ágban kezdtünk el fejleszteni. Lecseréltük a Windows Forms-ot C++ konzol alkalmazásra, melyekbe így beletudjuk integrálni az ASCII karakter megjelenítést. Illetve a felhasználó konzolos formában játszhatja végig a játékunkat.


MOSZE
Tesztelés
Program struktúrája
Windows Forms
C++ nyelv
C# nyelv
C++ command line
Dokumentáció
Folyamatos dokumentáció
Dokumentáció a végén
Megjelenítés
ASCII karakterekkel képek
Minden dialoghoz háttérnek
Történet
Egyetem kalandjáték
Mérföldkövek beleintegrálása
Képességek, melyeket a játék alakít

https://app.mindmup.com/map/new/1666609059200

Program

A program működése



1. A felhasználó elindítja a programot.

2. Megjelenik a kezdő story.
3. A megjelenő opciók közül választ egyet a felhasználó.
4. A kiválasztott opciót továbbítja a kontrollernek.
5. Ismétlődik a 2-4.sorozat a játék végéig.
6. Befejeződik a program.

A program osztályai

`class Story`

- Konstruktora egy link, ami a megnyitandó „.game” fájlt jelöli
- Attribútumai:
  - `string body`: a megjelenítendő szövegtörzs
  - `vector<string> btn_texts`: a megjelenítendő „gombok”, opciók
  - `vector<string> btn_links`: az opciókhoz tartozó linkek (a következő story)
- Metódusai:
  - `void open_link(string btn_link)`: a megadott link alapján megnyitja az adott fájlt – mindenképpen a `../story/[link].game` helyen keresi; beállítja a `btn_texts` és `btn_links` változókat

`class Game_Handler`

- Konstruktora egy `Story`, amit megjelenít
- Metódusai:
  - `void next_story(Story next_story)`: megjeleníti a `story.body` és a `story.btn_texts` szövegeket

A program segédfüggvényei

`namespace functions:`

`vector<string> split(string s, string delimiter)`: a bejövő `s` stringet felosztja a `delimiter` alapján, és a substringeket egy vektorban adja vissza

`void set_console_appearance()`: egy előre meghatározott módon beállítja a megjelenítő konzolt

- x helyzet: 100 px
- y helyzet: 100 px
- magasság: 800 px
- szélesség: 600 px
- betűtípus: `FF_DONTCARE`
- betűmagasság: 24 px

A main() függvény

- Inicializálja a lokális változókat: `option` és `link`
- Deklarálja a `link`-et a kezdőstoryra (`_start.game`)
- Meghívja a `set_console_appearance()` függvényt
- Létrehoz dinamikusan egy `s` storyt `link` alapján és egy `g` game handlert `s` alapján
- Bekéri a felhasználótól az `option`-t
- Átírja `link`-et a kiválasztott linkre: `btn_links[option]`
- Felszabadítja az `s` memóriaterületen lévő storyt, így megfékezve a memory leaket
- Törli a konzol képernyőjét: `system("CLS");`
- Felülírja `s`-t egy dinamikusan létrehozott story-val a `link` alapján
- Meghívja a `g` game handler `next_story` metódusát, így kiírja a következő story szövegét és opcióit
- Mindezeket megismétli egészen addig, amíg tart a játék, vagyis van következő link
- Felszabadítja a `g` memóriaterületet

Megvalósítandó feature-k

Implementálandó feladat az ASCII képek feldolgozására és megjelenítésére szolgáló osztály. Lehetséges módja ennek, hogy a Game_Handler osztályt módosítjuk, refaktoráljuk. Másik lehetséges módja egy teljesen különálló interfész, amit meghív a Game_Handler adott esetekben. Még eldöntendő, hogy a .visual fájlokat

A történet tárolása

Fájlformátum

Minden story elemet külön fájlban tárolunk a ./story mappában. A fájlok neve a következő: [story_name].game

Az elemek a következőképp épülnek fel:

[Többsoros szöveg,

“Body Text”]

|[opció1]>[link1]

|[opció2]>[link2]

(|[opció3]>[link3])

A többsoros szöveg, amit szabadon írunk. Ez lesz a body text később a programban, ezzel találkoznak a felhasználók először.

Az opciók a választási lehetőségek, amiket a felhasználó választhat. Ha az input 0, akkor az első opciót választja, ha 1, akkor a másodikat, ha 2, akkor a harmadikat.

Egy link vezethet egy .game vagy egy .visual fájlhoz. Ha az előbbi teljesül, akkor egy újabb story elemet kapunk, ha az utóbbi, akkor egy ASCII kép jelenik meg, amit billentyű megnyomásával tud továbbléptetni a felhasználó.

A .visual fájlnak nincs különleges felépítése, csak a megjelenítendő karakterláncot tartalmazza, amit egy külön controller osztály fog feldolgozni és megjeleníteni.

Generátor segédprogram

Segítségként írtunk egy story generátor programot, ami user inputból automatikusan elkészíti a megadott ".game" fájlokat. Beadjuk a fájl nevét, a story szövegét, az opciók szövegét és linkjét, majd ha üres stringet talál, előről kezdődik.

Hátránya, hogy a szövegekbe nem tudunk sortörést írni (egyelőre, megoldás későbbre: a \n karaktersor helyettesítése whitespace-el), illetve hogy csak a program megszakításával tudunk kilépni, mivel az egyszerűség kedvéért végtelen while ciklusban fut.