



A “JÁTÉK PROTOTÍPUSOK” TANTÁRGY ISMERETANYAGÁ- NAK KIDOLGOZÁSA ÉS AZ ELSŐ OKTATÁSI ÉV TAPASZTALATAINAK ÖSSZEGZÉSE

KISS ÁRON

Miskolci Egyetem
Informatikai Intézet

Alkalmazott Informatikai Intézeti Tanszék
aron.kiss@uni-miskolc.hu

NEHÉZ KÁROLY

Miskolci Egyetem
Informatikai Intézet

Alkalmazott Informatikai Intézeti Tanszék
karoly.nehez@uni-miskolc.hu

Absztrakt: A Miskolci Egyetem Informatikai Intézetében 2022 szeptemberében indult el a „Játék prototípusok” című tantárgy oktatása. A cikkben ismertetjük más egyetemek oktatási módszereit, a tantárgyunk célkitűzéseit, fontosabb témaköreit, majd bemutatjuk az oktatásához kiválasztott programozási keretrendszert, összehasonlítva azt más szoftverekkel. Az első év tapasztalatai alapján elmondható, hogy a hallgatók lelkesen fogadták a tananyagot és számos színvonalas beadandó feladatot készítettek. Tanulmányunk célja az első tapasztalataink bemutatása, illetve olyan javaslatok és tanulságok megfogalmazása, melyek segíthetnek más oktatókat a saját játékfejlesztési kurzusuk elkészítésében vagy továbbfejlesztésében.

Kulcsszavak: *játékfejlesztés, platformfüggetlen programozás, oktatás*

1. Bevezetés

A játékfejlesztés és számítógépes vizualizáció gyorsan fejlődő területek, melyek egyaránt hozzájárulnak a szórakoztató iparágak fejlődéséhez és az innovációhoz. A számítógépes vizualizáció nem csupán a játékipar számára fontos, szerepe van többek között az oktatásban, a tervezésben és az orvostudományban is. A szemléletes és érzékletes grafikák segítségével könnyebben megérthetők komplex folyamatok, a virtuális modellezés révén pedig lehetőség nyílik a valóságos környezetek előzetes vizsgálatára, illetve alternatívák kipróbálására.

A prototipizálás kritikus szerepet játszik a játékfejlesztés folyamatában. Prototípusok segítségével a fejlesztők hatékonyan tesztelhetik az ötleteiket, kísérletezhetnek különböző játékmekanikákkal és korai visszacsatolást kaphatnak a játékélményt illetően. A visszajelzések alapján a projekt kezdeti fázisában nyílik lehetőség a tervek finomítására. Ezen túlmenően prototipizálás segítségével korán kiszűrhetők a tervekben szereplő ellentmondások és csökkenthető a projekt kockázata. A prototípusok fejlesztése tehát nemcsak a fejlesztés hatékonyságát növeli, hanem a végső termék minőségét is javítja, és lehetővé teszi a fejlesztők számára, hogy rugalmasabban alkalmazkodjanak a változó igényekhez és elvárásokhoz.

A Miskolci Egyetem Informatikai Intézetében 2022 szeptemberében indult el a „Játék prototípusok” című tantárgy oktatása, mely a programtervező informatikus képzés 5. féléves, kötelezően választható tárgycsoportjában foglal helyet. A tantárgy meghatározó célkitűzése az, hogy a hallgatók elmélyült és átfogó ismereteket szerezzenek a játékfejlesztés alapvető fogalmaiban és a leggyakrabban alkalmazott koncepciók terén. Az oktatás emellett hangsúlyt fektet a játéktervezés alapelveire és az ezzel kapcsolatos eszközökre, hogy a tantárgyat teljesítő hallgatók rendelkezzenek a szükséges kompetenciákkal és tudással a sikeres játékfejlesztéshez, valamint kellő szintű alapismeretük legyen a későbbi, mélyebb tudást igénylő ráépülő tantárgyak teljesítéséhez.

Cikkünkben bemutatjuk a játékfejlesztés-alapú oktatás legfontosabb előnyeit, majd megvizsgáljuk a tantárgy szerepét és helyét a programtervező informatikus képzésben. Megvizsgáljuk más egyetemek oktatási gyakorlatait a témakörben, illetve bemutatjuk az új tantárgy tematikus felépítését. A cikk további részében ismertetjük azt a folyamatot, amely során kiválasztottuk az oktatáshoz szükséges keretrendszert, bemutatva azokat az elemzési szempontokat, amelyek alátámasztják a keretrendszer megfelelőségét a tantárgy célszerű és hatékony oktatásához. Ezt követően áttekintjük az első év oktatási tapasztalatait, valamint azokat az észrevételeket, tanulságokat és sikereket, melyekkel oktatóként a tantárgy első hallgatóival együtt szembesültünk.

2. Játékfejlesztés alapú oktatás

A szakirodalomban számos forrás található a játékorientált oktatás hatékonyságára. [17] a megfelelő játékfejlesztő keretrendszer megválasztásának szempontjait és fontosságát, valamint a játékfejlesztés alapú tanulás eredményességét vizsgálja, megállapítva, hogy a módszer eredményesebben javítja a diákok tudását, készségeit és általános hozzáállását a hagyományos frontális oktatáshoz képest. [15] megállapítja, hogy egy játékfejlesztési kurzus hatékonyan megvalósítható korlátozott anyagi források mellett is, valamint előnyösnek találja, ha az oktatásban az elméleti és gyakorlati ismeretek integrált formában jelennek meg, szorgalmazza továbbá a hallgatók kooperációján alapuló tanulást. A játékfejlesztés-alapú oktatási módszer más műszaki területen is sikeresnek bizonyul. [18] egy játékfejlesztési keretrendszer alkalmazásával foglalkozik szoftverarchitektúrák oktatása során. [16] programtervezési

minták játékfejlesztésen keresztüli oktatásának pozitív tapasztalatairól számol be. A vizsgált játékfejlesztés-alapú oktatási módszerek eredményeként a tanulók könnyebben megértették és alkalmazták a megfelelő szakmai fogalmakat és elveket. Ezen kívül a játékos környezet ösztönző hatása miatt nőtt az érdeklődés és a részvétel a tanulási folyamatban.

3. A tantárgy képzésben elfoglalt helye és célkitűzései

A „Játék prototípusok” a programtervező informatikus képzés kötelezően választható tárgycsoportjában felvehető tantárgy, melynek javasolt teljesítési ideje a képzés 5. féléve. A mintaterv szerinti ütemben teljesítő hallgatók ekkorra általános ismereteket szereztek legalább egy procedurális és egy objektumorientált programozási nyelvben, ismerik a szoftverek és szoftverrendszerek tervezésének általános elveit, valamint alapvető számítógépi grafikai ismeretekkel is rendelkeznek [1]. Fontos megjegyezni azonban, hogy a tantárgynak kötelező tanulmányi előfeltétele nincs, így a tananyagot és a technológiai alapozó órák mennyiségét a hallgatók valós tudásához szükséges igazítani. A tantárgy keretein belül kiemelt figyelmet kell fordítanunk a játékfejlesztés során felmerülő kihívásokra és az ezekre adott megoldásokra. A hallgatók a félév ideje alatt részletes betekintést nyerhetnek a játéktervezés alapelveibe, a karakterfejlesztés stratégiáiba, valamint a játékelmény felépítésének lépéseibe.

Az oktatás fókuszában nem elméleti ismeretek állnak, hanem gyakorlati alkalmazások kapnak kiemelt jelentőséget. A hallgatók lehetőséget kapnak arra, hogy a tantárgy keretein belül közvetlen tapasztalatokat szerezzenek a fejlesztési folyamat során alkalmazott különböző keretrendszerekkel és tervező alkalmazásokkal kapcsolatban. A tantárgy ennek érdekében gyakorlatorientált megközelítésben ismerteti meg a hallgatókat a játékfejlesztéshez használt keretrendszerek alapjaival és működésével, hogy a megszerzett tudás biztos alapot képezzen számukra a későbbi fejlesztési projektekből való részvételhez.

4. Más intézmények oktatási módszerei

A hazai egyetemek játékfejlesztési kurzusain alkalmazott oktatási módszerek vegyes képet mutatnak. A legtöbb intézmény esetén megtalálható olyan tantárgy, amely számítógépi grafikai fókuszú, és elsősorban matematikai, ábrázoló geometriai módszerekre összpontosít [19, 20, 21]. Elérhetőek továbbá olyan játékfejlesztési kurzusok is, melyek részletes, grafikai technológiákat, és azok implementációját mélyebb szinten tárgyaló órákat kínálnak, ideértve a játékok mögött rejlő olyan technikai aspektusokat, mint a renderelés, az animáció és az árnyalás [21]. A játékfejlesztési feladatokat egyes esetekben más fókuszú tantárgyakba is integrálják. Ez azt jelenti, hogy a hallgatóknak lehetőségük van arra, hogy más szakterületeken szerzett tudást felhasználva ismerkedjenek meg a játékfejlesztés alapjaival, pl. [22] szoftvertechnológiai és grafikus felhasználói interfészek tervezésével kapcsolatos ismeretek elmélyítéséhez írja elő játékprogram implementációját.

Nemzetközi egyetemeken tantárgyait megvizsgálva találkozhatunk kifejezetten játékok prototipizálásával foglalkozó kurzusokkal is. [23] korszerű játékfejlesztési platformokba nyújt betekintést, célja, hogy a hallgatók gyakorlati tapasztalatokat szerezzenek a játékfejlesztési folyamatokban, ipari szabványok alkalmazásában és különféle játékmotorok használatában. [24] keretében a hallgatók csapatokat alkotva szerezhettek tapasztalatot rapid prototípus építésben, ötleteik szóbeli prezentációjában és az elkészült prototípusok kiértékelésében.

5. A tantárgy tematikus felépítése és teljesítési követelményei

A tantárgy tematikáját úgy építettük fel, hogy olyan alapvető fogalmakra és koncepciókra összpontosítson, amelyek elengedhetetlenek a játékfejlesztés alapos megértéséhez, az alkalmazott fejlesztési keretrendszerrel függetlenül is. A tananyag összeállításánál a 2. fejezetben említett források mellett a [10, 11, 12, 13] könyveket használtuk támpontként a tartalom és a struktúra kialakításához. A meghatározott témakörök segítik a hallgatókat abban, hogy mélyreható és teljes körű tudást szerezzenek a tervezési és fejlesztési folyamat minden aspektusáról. Célunk az, hogy a hallgatók ne csak elméleti szinten ismerjék meg ezeket a fontos koncepciókat és alapfogalmakat, hanem gyakorlati példákon keresztül is el tudják sajátítani azok alkalmazási módjait valós játékfejlesztési projektek során.

A tantárgy ismeretanyaga a következő alapfogalmak, koncepciók és témakörök köré szerveződik:

Játéktervezési alapelvek. Olyan tervezési alapelvek és mintadokumentumok áttekintése, melyek segítségével megfelelő részletességgel dokumentálható és strukturálható a játékfejlesztés tervezett folyamata, valamint leírhatók a játék alapvető elemei, mechanikái és funkciói. A hallgatók így megérthetik egy játékfejlesztési projekt komplexitását, jobban felkészülve valós projektekben való részvételre. Ez segíthet továbbá abban is, hogy hatékonyabban kommunikáljanak a csapattársaikkal és könnyebben megértsék az ötleteiket.

Technológiai alapismeretek. Az oktatás alapjául szolgáló keretrendszerhez kapcsolódó programozási nyelv és technológiák áttekintése.

Sprite-ok kezelése, animációk, interakciók. A „sprite”-ok a képernyőn megjelenő 2D-s grafikai elemek a játékokban. A játékok fejlesztéséhez szükséges áttekintnünk ezek alapvető kezelését, animálását (pl. in-between animációk, spritesheet animációk), illetve a sprite-ok közötti egyes hatásokat (pl. átfedésvizsgálat).

Kamerakezelés, kameraeffektek. A megfelelő kamerakezelés és kameraeffektek segítenek maximalizálni a játékélményt, beleértve a játékos által tapasztalt látványos és élménydús jeleneteket, lehetővé teszik továbbá a játéktervezők számára, hogy irányítsák a játékos figyelmét, valamint meghatározzák a játékmenet dinamikáját.

Fizikai motor alkalmazása. A játékok sok esetben valós fizikai jelenségeket modelleznek, így a hallgatók számára előnyös ezen jelenségek kezelésének és alkalmazásának megértése. Ennek részeként a fizikai motor esetében áttekintjük a gravitáció, a rugalmas- és rugalmatlan ütközés, a gyorsulás, a testekre ható egyéb erők és további hétköznapi jelenségek játékokban történő szimulációját.

Tilemapek kezelése. A tilemapek lehetővé teszik térképek hatékony kezelését, valamint fontos szerepet játszanak a játékkörnyezet létrehozásában és a játékbeli navigáció kialakításában. Alkalmazásukkal részletes és sokszínű környezet hozható létre, ami növeli a játék vizuális vonzerejét és a játékélményt, strukturáltságuk pedig segíti a programozókat a játék továbbfejlesztése és karbantartása során.

A tárgy teljesítéséhez a hallgatóknak 2-3 fős csoportokat alkotva kell egy gyakorlati feladat megoldásával bizonyítaniuk az órákon bemutatott témakörök ismeretét. Ehhez a gyakorlatok keretében használt eszközök segítségével egy 2 dimenziós, ügyességi, logikai vagy platformer játékot kell megtervezniük és implementálniuk. A tervezési és fejlesztési folyamat eredményeit verziókezelő rendszerben kell közzétenniük, valamint az utolsó oktatási héten demonstrálniuk kell az elkészült játék funkcióit.

6. Az oktatott keretrendszer kiválasztásának szempontjai

Az oktatáshoz használt keretrendszer kiválasztásakor több tényezőt vettünk alapul a döntés meghozatalához. Az egyik legfontosabb szempont az volt, hogy olyan platformfüggetlen megoldást találjunk, amely lehetővé teszi a tanulást bármelyik operációs rendszeren. Továbbá olyan rendszert kerestünk, amely ideális esetben általánosan, de legalább akadémiai licenccel ingyenesen elérhető. Meghatározó szempontként vettük figyelembe továbbá a kiforrottságot, hiszen a célunk az volt, hogy a kiválasztott keretrendszer a tantárgyi tematika minden részét le tudja fedni. A dokumentáció minősége és a rendelkezésre álló példaprojektek, valamint oktatóanyagok szabad hozzáférése szintén fontos szerepet játszott döntésünkben, így olyan könyvtárat kerestünk, amely megfelelő színvonalú, jól dokumentált, és széles körű támogatással rendelkezik, lehetőséget biztosítva az otthoni tanulásra, és a tananyagon túli további témakörök megismerésére.

7. A megvizsgált szoftverek összehasonlítása

Az oktatás alapjául az előzőekben ismertetett szempontoknak megfelelő keretrendszert, könyvtárat vagy standard API-t szerettünk volna választani. A cross-platform működés fontossága miatt webes technológiákat tekintettünk át.

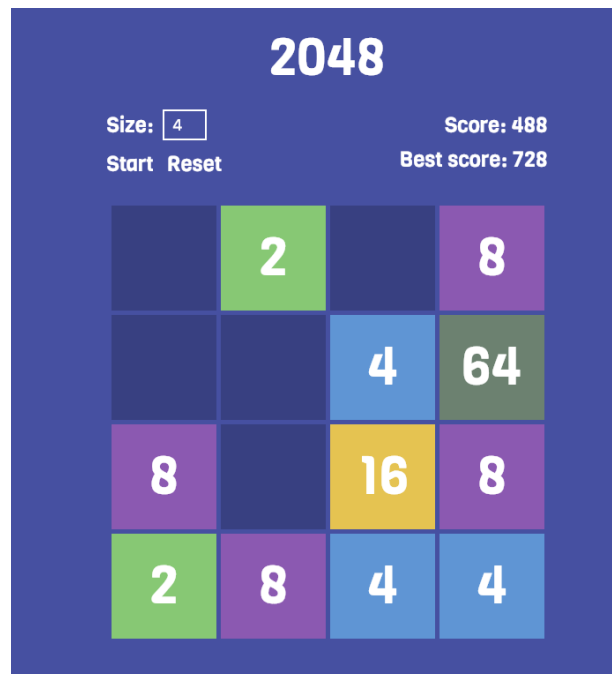
Az 1. táblázatban ismertetjük a vizsgált technológiai megoldásokat, melyek mindegyike részben eltérő megközelítés szerint támogatja a játékfejlesztés folyamatát.

1. táblázat*Az oktatáshoz használt technológia kiválasztásának szempontjai [2, 3, 4, 5]*

	HTM5 Canvas API	Kontra.js	p5.js	Phaser
Első kiadás dátuma	2007-05-01	2015-05-17	2014-07-21	2013-04-12
Utolsó kiadás dátuma	2023-10-12	2023-01-27	2023-07-10	2023-04-12
Platform-független	Igen	Igen	Igen	Igen
API szintje	Alacsony	Magas	Magas	Magas
Követők száma	N/A	858	20 034	35 364
Kamera-kezelés	Nincs	Van	Van, több kamera	Van, több kamera
Animáció	Alacsony szintű	Spritesheet	Magas szintű, nincs sprite-kezelés	In-betweening Spritesheet
Fizikai motor	Alacsony szintű	Nincs	Nincs	Van
Tilemap-támogatás	Nincs	Van	Nincs	Van
Licenc	–	MIT	GNU LGPL	MIT

A *Canvas* API a HTML nyílt webes szabvány része, amely lehetővé teszi a fejlesztők számára, hogy különböző grafikai elemeket, például vonalakat, görbéket, szövegeket és képeket jelenítsenek meg, interaktív vizuális élményt létrehozva a webes alkalmazásokban. A *Canvas* API széles körben használatos a játékfejlesztésben, adatvizualizációban és egyéb webes projektekben.

Az 1. ábrán a 2048 játék egy *Canvas* API-alapú implementációja látható. A projekt alkalmazza többek között a geometriai formák megjelenítését, azok stílusának testreszabását, valamint a felhasználói bemenet kezelését.

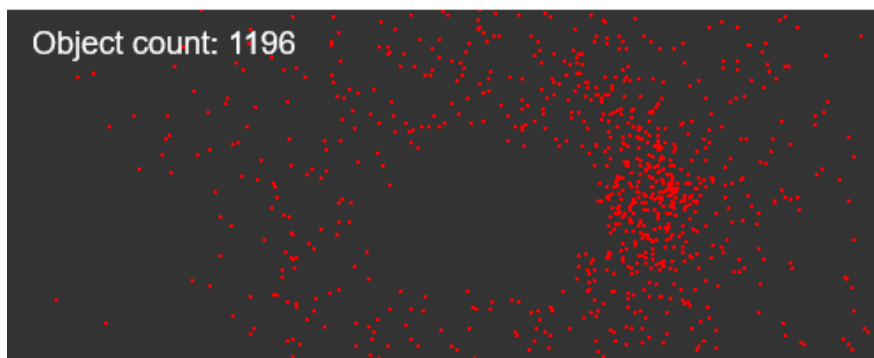


1. ábra. A „2048” játék egy Canvas API-alapú implementációja [7]

A *Canvas* API legfontosabb előnye széles körű támogatottsága és részletes dokumentációja, valamint az elérhető oktatóanyagok jelentős száma. Az oktatásban való felhasználhatósága azonban korlátokba ütközik alacsony szintű interfészei miatt. A többi vizsgált keretrendszer a háttérben alkalmazza ugyan a *Canvas* API-t, de olyan, magas szintű absztrakciókat biztosítanak a programozók számára, melyek jelentősen megkönnyítik a játékfejlesztést. A „Játék prototípusok” tantárgy elsődleges célja nem a mélyreható ismeretek átadása, melyekkel a játékok legkisebb funkciói is részletesen kidolgozhatók és optimalizálhatók, hanem az alapvető tervezési és fejlesztési koncepciók megismertetése, melyet a *Canvas* API alacsony absztrakciós szintje nem támogat megfelelően.

A *Kontra.js* egy „light-weight” JavaScript könyvtár, amely segíti a fejlesztőket a HTML5 *Canvas* API hatékonyabb használatában a játékfejlesztés során. A könyvtár egyszerűséget és gyors tanulást tesz lehetővé, beépített funkciókkal és kényelmi eszközökkel. Optimalizált és egyszerűsített funkciókat kínál a játékfejlesztők számára anélkül, hogy felesleges terhet róna a rendszerre vagy a fejlesztési folyamatra. A könnyűsúlyú jelleg lehetővé teszi a gyors betöltést és futást, minimalizálva ezzel a rendszer erőforrásainak felhasználását. A 2. ábrán egy olyan alkalmazás látható, mely a könyvtár „object pooling” funkcióját teszteli. Ennek lényege a memóriába már betöl-

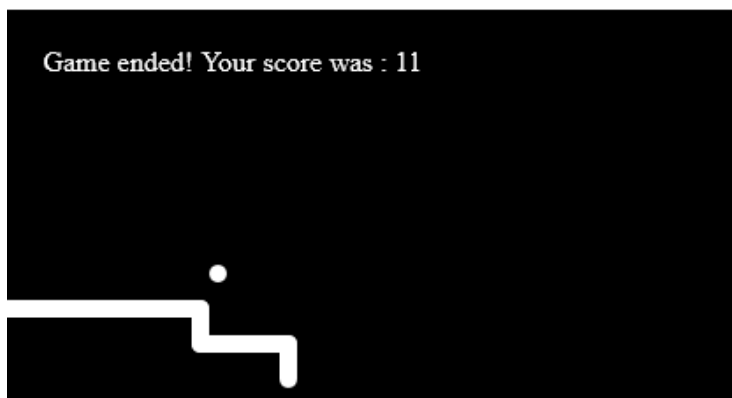
tött, de továbbiakban nem használt sprite-ok újrafelhasználása, a teljesítmény növe-
lése és a memóriafelhasználás csökkentése érdekében. Ez ideális lehet például ré-
szecske-rendszerek és „Shoot 'em Up” stílusú játékok vizualizációja esetén.



2. ábra. *Object pooling tesztelése Kontra.js-ben [9]*

A *Kontra.js* a *Canvas* API-nál magasabb szintű, jól strukturált API-t biztosít a prog-
ramozók számára, azonban a hangsúlyt a teljesítmény optimalizációjára helyezi, és
nem fedí le teljes mértékben a tantárgy tematikus felépítését. Ezen témakörök teljes
körű bemutatásához további kiegészítő csomagok megismerésére lehet szükség.

A *p5.js* egy JavaScript könyvtár, amely kreatív vizualizációk készítését segíti webes
alkalmazásokban. Képes egyszerű és összetett grafikák és animációk létrehozására a
játékokban, beleértve a karaktereket, a környezetet és a hatásokat is. Emellett lehetővé
teszi a fejlesztők számára a játékosokkal való interakció kialakítását, javítva ezzel a
játékelményt. A hangkezelésre is lehetőséget nyújt, ami lehetővé teszi a hanghatások
és a zenei elemek beépítését a játékba. A 3. ábrán egy egyszerű snake játék imple-
mentációja látható, mely egyebek mellett igényli a geometriai alakzatok kirajzolását,
a felhasználói bemenet kezelését, valamint a játéktér képkockánkénti frissítésének
megvalósítását.



3. ábra. Snake játék p5.js felhasználásával implementálva [8]

Tapasztalataink szerint a p5.js egy vizualizációs célokra kitűnően használható, intuitívan is könnyen felfedezhető könyvtár, azonban oktatási célú alkalmazhatóságát nagyban befolyásolja a beépített fizikai motor és tilemap-támogatás hiánya. Az animációk terén széles körű funkcionalitással rendelkezik ugyan, azonban ezek nem a játékfejlesztést helyezik előtérbe, in-between és spritesheet animációk létrehozására nem kínál lehetőséget. Ezen funkciók külső könyvtárak segítségével ugyan elláthatók lennének, de a célunk az volt, hogy elkerüljük a körülményes telepítési és konfigurációs lépéseket, és elsősorban a tantárgyi tematika elemeire fókuszáljunk a tárgy oktatása során.

A Phaser 3 egy teljes körű játékfejlesztési keretrendszer, amely lehetővé teszi teljes játékok létrehozását HTML5 környezetben. A keretrendszer teljesítményorientált architektúrával rendelkezik, biztosítva a jó minőségű animációkat és a magas színvonalú játékelményt. Kiterjedt referenciadokumentáció és aktív közösségi támogatás jellemzi. A 4. ábrán a Phaser fizikai motorjának hibakeresési módja látható, mely az egyes sprite-ok határait, és a játékobjektumokra ható erőrendszer eredőjét szemlélteti.



4. ábra. A Phaser 3 fizikai motorjának hibakeresési módja [6]

A Phaser 3 kimondottan játékfejlesztési célú keretrendszer, mely funkciók széles körét biztosítja a programozók számára. A kamerarendszer-kezelés és a sprite-animációk mellett megtalálható benne a fizikai jelenségek szimulációjához szükséges motor-, valamint a legnépszerűbb tilemap-formátumok támogatása is. Ezen funkciók alapján a vizsgált szoftverek közül a Phaser 3 bizonyult a legalkalmasabb megoldásnak a tantárgyi tematika lefedésére. A Phaser melletti további érv a referenciadokumentáció

részletessége, az elérhető segédanyagok nagy száma, valamint a jelentős létszámú fejlesztői és támogatói közösség.

8. Hallgatói visszajelzések és oktatási tapasztalatok

A tantárgy első oktatásban töltött féléve alatt számos tapasztalatot szereztünk, melyek alapján a kontaktórák felépítése tovább fejleszthető. Tapasztalataink szerint a hallgatók JavaScript ismeretei hiányosak, így ennek fejlesztésére, illetve a korábban tanultak felidézésére nagyobb hangsúlyt kell fektetni. A hallgatók többsége erősen típusos programozási nyelvekben (jellemzően C, C#, Java) szerzett korábban rutint, amelyek után a JavaScript (JS) dinamikus típusossága szokatlanul hat számukra. A dinamikus típusosság mellett szokatlan újdonság továbbá az aszinkron működés kezelése esemény-orientált módon, valamint „callback” függvények segítségével, melyek oktatására a technológiai alapok ismertetése során kiemeltebb figyelmet kell fordítani.

A *Phaser* használata során kisebb kellemetlenséget okozott továbbá a kódkiegészítés hiánya, mely a keretrendszer JS-ben írt kódbázisára vezethető vissza. Ez a TypeScript (TS) programozási nyelv részleges bevezetésével orvosolható. A gyakorlatokon bemutatott példák teljes mértékben TS-ben történő implementálása olyan konfigurációs és programozástechnikai megoldásokat igényelve, melyek a játékprogramozás helyett túlzottan a webfejlesztés, illetve egy új programozási nyelv megtanulásának irányába mozdítanak el a tárgy fókuszát. Ehelyett, tanulva a tavalyi év tapasztalataiból, idén a *Phaser* által biztosított TS-alapú típusdefiníciókat használjuk a kódkiegészítés konfigurálásához, a JS-ben történő programozás megtartása mellett.

A hallgatók a félév során aktívan és lelkesen foglalkoztak a kiadott feladatokkal, rendszeresen kérdeztek, akár a kötelező tananyagon túlmutató témakörökben is. Ezen kérdések és visszajelzések alapján érdemes fontolóra venni többek között az okostelefonokra történő játékfejlesztés kiegészítő tananyagként történő bevezetését, így egy videójáték-piacon meghatározó részesedéssel rendelkező területen bővítve a hallgatók tudását [14]. A *Phaser* cross platform keretrendszerként alapvetően támogatja a mobilalkalmazásokra történő fejlesztést. *Apache Cordova* vagy *Capacitor* alkalmazásával a különböző platformok alkalmazásboltjaiban is publikálható, natív mobilalkalmazások lennének fejleszthetők, a JS-alapú kódbázis megtartása mellett.

A hallgatók munkájának eredményeként jó színvonalú beadandó feladatok születtek, melyek egyikét mutatja be az 5. ábra. A képen látható alkalmazás egy sakkjáték, melynek vektorgrafikus figuráit a hallgatók maguk rajzolták, a *Phaser*-be épített grafikus interfész segítségével. A sakkjáték szabályainak implementációjához külső könyvtárat nem használtak fel. A játék menetét hanghatásokkal és animációkkal tették teljessé. Olyan többfelhasználós játékot alkottak, amely virtuális sakkórát biztosít, az időmérés automatikus támogatásához.

A tárgy első féléves eredményeinek áttekintése alapján elmondható, hogy azon hallgatók, akik a tantárgy kontaktóráinak legalább 50%-án megjelentek, sikerrel teljesítették a kitűzött elvárásokat. Bizakodásra ad okot továbbá, hogy az idei évben megközelítőleg 40%-kal emelkedett a tárgyat felvett hallgatók létszáma.



5. ábra. Kétfelhasználós sakkjáték időméréssel Phaser 3-ban (Molnár Balázs és Szöllősi János munkája).

9. Összefoglalás

A cikkben bemutatott Miskolci Egyetemen oktatott „Játék prototípusok” című tantárgy kidolgozásának szempontjait, valamint összegeztük az oktatás első tapasztalatait, kitérve a tantárgy további fejlesztésének lehetőségeire.

A hazai és nemzetközi egyetemek játékfejlesztési kurzusain alkalmazott oktatási módszerek nagyon változatosak. A saját tantárgyunk tematikáját a játékok fejlesztése során alkalmazott alapvető fogalmakra és koncepciókra építettük, elméleti és gyakorlati jellegű ismereteket ötvözve.

A tantárgyban fontos szerepet kapnak a játéktervezési alapelvek, a technológiai ismeretek, a sprite-ok kezelése, a kamerakezelés, a fizikai motor alkalmazása, és a tilemapok kezelése. Ismertettük továbbá a tárgy oktatásának alapjául szolgáló keretrendszer kiválasztását is, melynek során a platformfüggetlenség, az ingyenesség, a kiforrottság és a dokumentáció minősége voltak fontos szempontok. Négy népszerű játékfejlesztési- és vizualizációs eszközt (*Canvas API*, *Kontra.js*, *p5.js*, *Phaser 3*) hasonlítottunk össze, és bemutatottuk az ennek során szerzett tapasztalatainkat. A tárgy

oktatási célkitűzései alapján a *Phaser 3* keretrendszer bizonyult a legalkalmasabbnak az oktatás alapjául.

A tantárgy oktatása során szerzett tapasztalatokra alapozva megállapítottuk a kontaktórák fejlesztésének lehetőségeit, melynek egyebek mellett részét képezi a JavaScript ismeretek mélyebb megalapozása, a *Phaser 3* keretrendszerrel kapcsolatban felmerült technikai kihívások kezelése, valamint a mobileszközökre történő játékfejlesztés kiegészítő tananyagként történő bevezetése. A hallgatók a félév során aktívak voltak, és sikeresen teljesítették a kitűzött feladatokat, a tárgy iránti érdeklődés az idei évben növekedést mutatott.

Irodalomjegyzék

- [1] *Összesített tantárgyi háló, GE-BP*. Miskolci Egyetem Gépészmérnöki és Informatikai Kar. http://gepesz.uni-miskolc.hu/content/32/32_196.pdf (Hozzáférés: 2023. 10. 14.)
- [2] *straker/kontra*. GitHub. <https://github.com/straker/kontra> (Hozzáférés: 2023. 10. 14.)
- [3] *processing/p5.js*. GitHub. <https://github.com/processing/p5.js> (Hozzáférés: 2023. 10. 14.)
- [4] *photonstorm/phaser*. GitHub. <https://github.com/photonstorm/phaser/> (Hozzáférés: 2023. 10. 14.)
- [5] *HTML Standard*. W3C. <https://html.spec.whatwg.org/> (Hozzáférés: 2023. 10. 14.)
- [6] *Phaser – Examples – Custom debug colors*. <https://phaser.io/examples/v3/view/physics/arcade/custom-debug-colors> (Hozzáférés: 2023. 10. 14.)
- [7] *janmager/2048-canvas*. GitHub. <https://github.com/janmager/2048-canvas> (Hozzáférés: 2023. 10. 14.)
- [8] *p5.js – Examples*. <https://p5js.org/examples/interaction-snake-game.html> (Hozzáférés: 2023. 10. 14.)
- [9] *Kontra.js – Pool*. <https://straker.github.io/kontra/api/pool> (Hozzáférés: 2023. 10. 14.)
- [10] Geary, D. H. (2012). *Core HTML5 canvas. Graphics, Animation and Game Development*. Prentice Hall. Philadelphia, PA. ISBN 978-0132761611.
- [11] Seidelin, J. (2014). *HTML5 games (2nd ed.). Creating Fun with HTML5, CSS3, and WebGL*. John Wiley & Sons. Nashville, TN. ISBN 978-1-118-85545-4.
- [12] Feronato, E. (2018). *HTML5 cross platform game development using Phaser 3*. Self-published. Online. <https://triqui.gumroad.com/l/odKJf> (Hozzáférés: 2023. 10. 14.)
- [13] Leung, T. (2020). *Infinite Runner in Phaser 3 with TypeScript*. Ourcade. Online. <https://ourcade.co/books/infinite-runner-phaser3/> (Hozzáférés: 2023. 10. 14.)
- [14] Wijman, T. (2022). The Games Market in 2022: The Year in Numbers. <https://newzoo.com/resources/blog/the-games-market-in-2022-the-year-in-numbers> (Hozzáférés: 2023. 10. 15.).

-
- [15] Ritzhaupt, A. D. (2009). Creating a Game Development Course with Limited Resources. *ACM Transactions on Computing Education*, 9 (1), pp. 1–16.
<https://doi.org/10.1145/1513593.1513596>
- [16] Gestwicki, P., Sun, F. (2008). Teaching Design Patterns Through Computer Game Development. *Journal on Educational Resources in Computing*, 8 (1), pp. 1–22.
<https://doi.org/10.1145/1348713.1348715>
- [17] Wu, B., Wang, A. I. (2012). A Guideline for Game Development-Based Learning: A Literature Review. *International Journal of Computer Games Technology*, pp. 1–20.
<https://doi.org/10.1155/2012/103710>
- [18] Wu, B., Wang, A. I., Strøm, J.-E., Kvamme, T. B. (2009). An Evaluation of Using a Game Development Framework in Higher Education. In: *2009 22nd Conference on Software Engineering Education and Training. IEEE*.
<https://doi.org/10.1109/cseet.2009.9>
- [19] DE. *Programtervező informatikus alapképzési szak képzési és kimeneti követelményei*. https://inf.unideb.hu/sites/default/files/upload_documents/2022_pti_bsc_tematikak_2021mt.pdf (Hozzáférés: 2023. 12. 10.).
- [20] BME VIK. Számítógépes grafika. <https://portal.vik.bme.hu/kepzes/targyak/VIIIA/B07/> (Hozzáférés: 2023. 12. 10.).
- [21] BME VIK. Játékfejlesztés. <https://portal.vik.bme.hu/kepzes/targyak/VIIIM289/> (Hozzáférés: 2023. 12. 10.).
- [22] ÓE. *Szoftvertchnológia és Grafikus Felhasználói Interfész Tervezése – Házi feladat követelmények*. https://users.nik.uni-obuda.hu/prog3/_progtools/prog4_game_requirements.pdf (Hozzáférés: 2023. 12. 10.).
- [23] Monash University. *FIT2145 – Game prototyping*. <https://handbook.monash.edu/2024/units/FIT2145> (Hozzáférés: 2023. 12. 10.).
- [24] Victoria University of Wellington. *CGRA 259 – Game Prototyping – Programming*. <https://www.wgtn.ac.nz/courses/cgra/259/2023/offering> (Hozzáférés: 2023. 12. 10.).