# Bevezetés

A modern számítógépes játékpiac rohamos fejlődése mellet a játékfejlesztők folyamatosan keresik azokat az innovatív megoldásokat, amelyekkel újszerű és lenyűgöző játékélményt hozhatnak létre. Az oldalnézetes (side-scroller) platformer játékok, amelyek az elmúlt évtizedekben jelentős népszerűségre tettek szert, különleges teret biztosítanak a kreativitás és a technológia találkozásának. Azonban ezeknek a játékoknak a fejlesztése komplex kihívásokkal jár, különösen a pályatervezés tekintetében, ahol a fejlesztőknek egyensúlyt kell találniuk az innováció, a játékélmény és a fejlesztési erőforrások között.

A szakdolgozatomban kettős célt tűztem ki: egyrészt egy teljesen működő, játszható 2D platformer játék tervezése és implementálása a Unity keretrendszerben, C# programozási nyelven, másrészt egy hozzá kapcsolódó pályagenerátor algoritmus fejlesztése, amely képes automatikusan, a felhasználó preferenciáit alapul véve változatos és kihívást jelentő pályákat létrehozni. Ez a kettős megközelítés lehetővé teszi, hogy nem csak elméleti síkon vizsgáljuk a pályageneráló algoritmusokat, hanem valós játékkörnyezetben is teszteljük azok hatékonyságát és hatását a játékélményre.

A szakdolgozatom kiterjed a platformer játék fejlesztésének minden aspektusára, beleértve a játékmechanika megtervezését, a grafikai elemek integrálását, valamint a felhasználói interfész megvalósítását. Mindezek mellett a fő hangsúly a pályagenerátor algoritmuson van, amely a játék alapvető részét képezi. Az algoritmus tervezésekor különös figyelmet fordítok a paraméterezhetőségre és az adaptivitásra, hogy a generált pályák ne csak változatosak és kihívást jelentőek legyenek, hanem jól illeszkedjenek a játék dinamikájához és stílusához.

A szakdolgozat során a platformer játék fejlesztési folyamatának minden lépését alaposan dokumentálom, a kezdeti koncepciótól a végleges implementációig. Ezen túlmenően, az algoritmus tervezése és implementációja során részletesen bemutatom a különböző programozási kihívásokat, a paraméterezési stratégiákat, és azokat a tesztelési módszereket, amelyekkel az algoritmus teljesítménye és a generált pályák játékbeli hatékonysága értékelésre kerül.

A szakdolgozatom így nem csak egy konkért algoritmus kidolgozására vállalkozik, hanem hozzájárul a videójáték fejlesztés megismeréséhez is.

# Az oldalnézetes játékok, valamint a pályageneráló algoritmusok jellemzői

## A fejezet célja

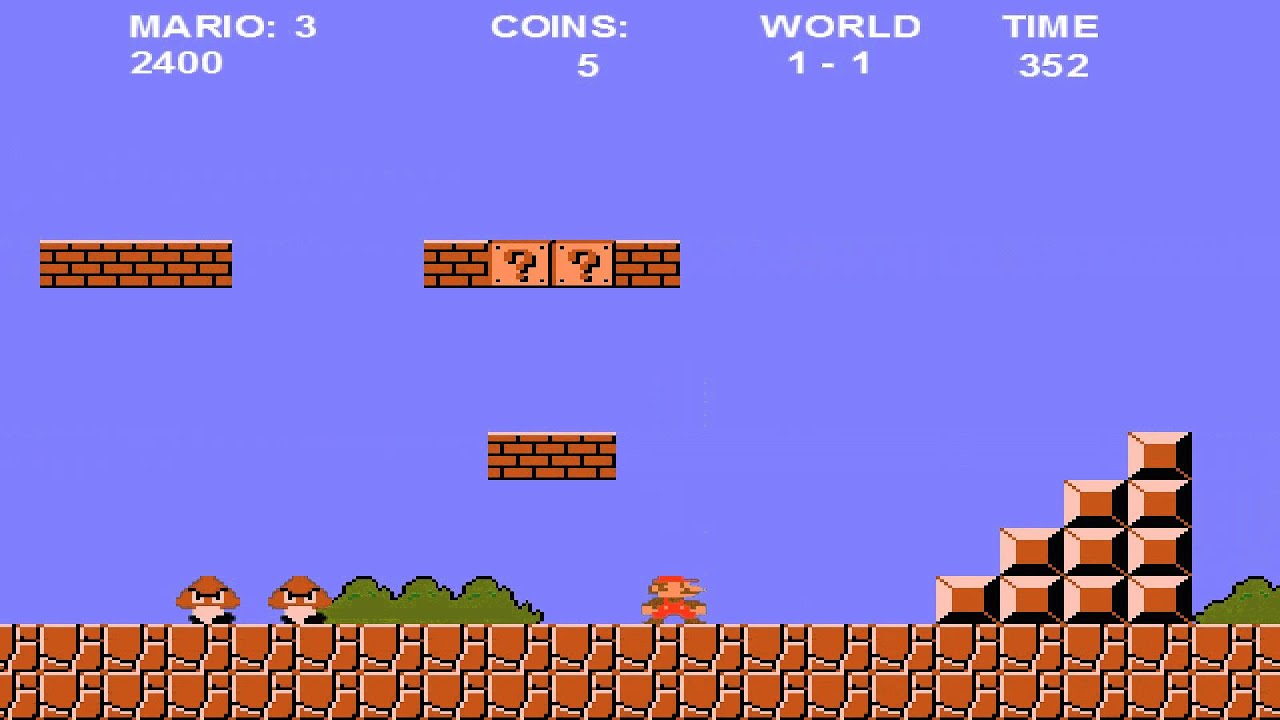
Ebben a fejezetben arra fogok törekedni, hogy részletesen bemutassam az oldalnézetes (side-scroller) játékok és a pályageneráló algoritmusok világát. Megvizsgálom az oldalnézetes játékok történelmi fejlődését, általános jellemzőit, és azt, hogy hogyan kapcsolódnak ezek a játéktípusok a pályageneráláshoz.

Ezen felül részletezni fogom a pályageneráló algoritmusok különböző típusait, módszereit, előnyeit és hátrányait.

## Az oldalnézetes játékok

### Fejlődésük

A 2D-s platformjátékok fejlődése gazdag és változatos utazás a videójátékok történetében. Egy rövid áttekintést szeretnék adni a fejlődésükről:

1. **A kezdetek:** Ez a műfaj a „Space Panic”-kel (1980) kezdődött, de a „Donkey Kong” (1981) volt az, amely a létrák és az ugrálás kombinálásával igazán megteremtette a mércét. Ezek a korai játékok többnyire egyképernyős platformerek voltak.
2. **„Side-scroller” korszak:** A „Super Mario Bros.” (1985) forradalmasította a műfajt a „side-scroller” pályákkal, emlékezetes karaktereket, „power-up”-okat és titkos útvonalakat vezetve be. Ebben a korszakban olyan játékok is megjelentek, mint a „Mega Man” és a „Metroid”, amelyek ezt a típusú játékstílust más elemekkel, például lövöldözéssel és felfedezéssel vegyítették.
3. **A technológia fejlődése:** A technológia fejlődésével a játékok elkezdtek pszeudo-3D elemeket tartalmazni. Az 1990-es években az olyan játékok, mint a „Crash Bandicoot” a platformer koncepciókat valódi 3D-s környezetbe helyezték.

2.2.1.1. ábra A Super Mario Bros. játékmenete

1. **A 16 bites korszak:** A „Mega Man X” és a „Donkey Kong Country” figyelemre méltó példái ennek az időszaknak. A 16 bites konzolok bevezetése lehetővé tette a feljavított párhuzamos görgetést és a részletesebb sprite-okat.
2. A képen rajzfilm, Számítógépes játék, játék, Stratégiai videojáték látható

   Automatikusan generált leírás**Korai 3D korszak:** A korai 3D platformerek közé tartoztak a 2,5D-s címek és a 3D-s perspektívájú, de 2D-s grafikájú platformerek. Az olyan játékok, mint a "Crash Bandicoot", lineáris pályákon maradtak, de vegyítették a járműveket és a másodpercek töredékeiben történő platformozást.

2.2.1.2. ábra A Crash Bandicoot játék

1. A képen szöveg, köd, képernyőkép, fekete látható

   Automatikusan generált leírás**Az indie újjáéledés:** A 2000-es évek végén és a 2010-es években az indie fejlesztők jelentős szerepet játszottak a 2D-s platformjátékok újjáélesztésében, és inkább a történetre és az innovációra összpontosítottak. Az olyan játékok, mint a "Braid", a "Limbo" és a "Super Meat Boy" egyedi mechanikájukkal és narratívájukkal mutatták be ezt a trendet.

2.2.1.3. ábra A Limbo játék

1. **Modern korszak:** A 2D platformjátékok az utóbbi években továbbra is népszerűek, gyakran a hagyományos játékmenetet modern tervezési elvekkel ötvözik. Az olyan címek, mint a "New Super Mario Bros." sorozat és a különböző indie játékok élénk és változatos műfajt tartanak fenn.

### Általános jellemzőik

A platformer játék, más néven platform videojáték, egy olyan játéktípus, amely jellemzően kétdimenziós grafikával rendelkezik, és amelyben a játékosok a képernyőn különböző platformokon ugráló vagy mászkáló karaktereket irányítanak.

A platformjátékokban egy karakter egy pályán navigál, hogy feladatokat teljesítsen, magas pontszámokat érjen el, vagy egyszerűen csak életben maradjon. Mivel ez a játékműfaj az évek során jelentősen megváltozott, sok ilyen játéknak más lehet a látványvilága. A következő jellemzők azonban gyakran megtalálhatók a platformjátékokban.

1. **Interaktív környezet:** Azt, hogy egy karakter mit tehet egy játékban, nagyban befolyásolja a szint vagy a környezet kialakítása. A platformjátékok célja különösen az, hogy próbára tegyék a játékost, miközben a főhőst olyan összetett akadályok elé állítja, mint a szöges platformok, halálos csapdák vagy lávával esetleg vízzel teli szakadékok.
2. **Third-Person nézőpont:** A játékos által irányított karakter az előtte lévő képernyőn látható, mivel sok platformjátékot úgynevezett „third-person” perspektívából készítenek.
3. A képen rajzfilm, Animációs film, clipart látható

   Automatikusan generált leírás**Vízszintes és függőleges mozgás:** A platformjátékok többsége kétdimenziós „side-scroller” játék, ami azt jelenti, hogy a játékos oldalról látja a karakterét, miközben a képernyő vízszintesen vagy függőlegesen mozog vele együtt.

2.2.2.1. ábra Horizontáls és vertikális mozgás

1. **Az ugrás kontrollálása:** A játékos irányítja a karakter ugrási képességét, ami a platformjátékok egyik fő szempontja. Ezekben a játékokban az ugrás gyakran szükséges a környezetben való mozgáshoz és a következő szintre jutáshoz.
2. **Történetmesélés és világépítés:** Bár a korai platformjátékokban nem volt ennyire elterjedt, a modern 2D-s platformjátékok gyakran tartalmaznak gazdag történetmesélést és részletes világépítést a játékélmény fokozása érdekében.

Ezek a funkciók együttesen hozzák létre azt az egyedi és gyakran kihívást jelentő élményt, amely a 2D platformer játékokat jellemzi. A műfaj az évek során jelentősen fejlődött, és minden játék a maga újításait és fordulatait vezette be ezekbe az alapvető összetevőkbe.

### A pályageneráló algoritmusok kapcsolódása a platformer játékokhoz

A számítógépes játékokban sokszor sokkal több tartalom megjelenítését szeretnénk elérni, mint amennyit valójában elő tudunk állítani vagy el tudunk tárolni. Vagy a tartalom előállítása során felmerülő korlátozások miatt - pl. egy kis gyártócsapat esetében -, vagy a tartalom tárolása, esetleg forgalmazása miatt.

Ezt azonban megkerülhetjük a procedurális generálással. Ez az, amikor a játék menetközben, játékidőben generál új tartalmat, ahelyett, hogy csak a korábban előállított tartalmat használná fel. Ha jól csináljuk, ez gyakorlatilag korlátlan tartalmat biztosíthat a játékunkban, sokkal alacsonyabb előzetes előállítási költségekkel.

A 2D-s platformjátékokban a térképgeneráló algoritmusok döntő szerepet játszanak a dinamikus és magával ragadó játékkörnyezetek létrehozásában. Ezek az algoritmusok generálhatnak térképeket előre létező szakaszok összerakásával vagy változó terepviszonyokkal rendelkező tájak rajzolásával. Egy gyakori módszer egy alapvonal megrajzolása (amely a talajt jelképezi), majd annak a magasságának a tájban való megváltoztatása a változatosság megteremtése érdekében.

Ezeknek az algoritmusoknak a 2D platformerekben való használata lehetővé teszi egyedi, procedurálisan generált világok létrehozását, ami növeli az újrajátszhatóságot és a játékosok érdeklődését. Minden egyes játékmenet más-más élményt nyújthat, a tájak az egyszerű és lapostól a komplex és többszintesig terjedhetnek.

Több játék is nagyszerűen használta az procedurális generálást. Például a Roguelight című játék bemutatja, hogy a procedurális generálással hogyan lehet mélyebb és sötétebb környezetet létrehozni, amely minden egyes játékmenettel változik.

2.2.3.1. ábra A Roguelight játék

A képen képernyőkép, szöveg látható

Automatikusan generált leírás

A képen Grafikus tervezés, szöveg, képernyőkép, poszter látható

Automatikusan generált leírásA "Diskophoros" egy másik érdekes cím, amely a gyors tempójú multiplayer akciót procedurálisan generált pályákkal kombinálja, így minden egyes játékmenet során új élményt tud nyújtani a játékosok számára.

2.2.3.2. ábra A Diskophoros játék

Ezek a példák szemléltetik a procedurális térképgenerálás változatos alkalmazásait a 2D-s platformjátékokban, jelentősen hozzájárulva a játéktervezéshez és a játékosok általi érdeklődés növeléséhez.

## Pályageneráló algoritmusok

Ebben a fejezetben a különböző generálási módszerekről, valamint a pályageneráló algoritmusok előnyeiről és hátrányairól fogok részletes leírást adni.

### Különbőző generálási módszerek

[Understanding Perlin Noise (adrianb.io)](https://adrianb.io/2014/08/09/perlinnoise.html)

[Perlin noise - Wikipedia](https://en.wikipedia.org/wiki/Perlin_noise)

[A slightly modified implementation of Ken Perlin's improved noise that allows for tiling the noise arbitrarily. (github.com)](https://gist.github.com/Flafla2/f0260a861be0ebdeef76)