1. Analízis modell

Dokumentum verzió: 1.0

Kiadás dátuma: 2025.11.16

Készítette: Infinite Loopers fejlesztőcsapat

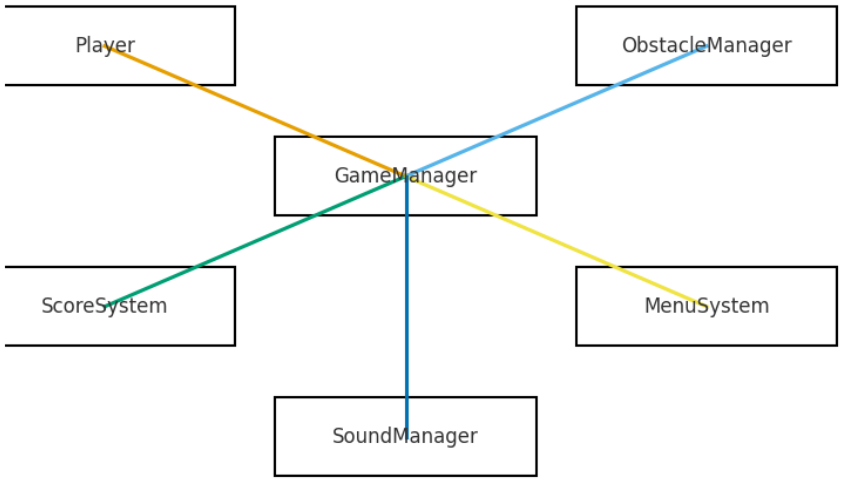
1. Bevezetés

Az analízis modell célja, hogy az Infinite Loopers játék szerkezetét és működését objektumorientált szemléletben mutassa be. A dokumentum arra fókuszál, hogy a H3-ban meghatározott funkcionális követelményeket (endless runner játékmenet, pontszámkezelés, menürendszer, Game Over logika) hogyan fordítjuk le konkrét osztályokra és azok kapcsolataira. A modell segíti a fejlesztőket abban, hogy közös képet alkossanak a doménről, és a későbbi rendszertervezés, implementáció során egyértelműen tudjanak hivatkozni a szereplőkre, felelősségekre és együttműködésekre.

2. Kezdeti osztálydiagram és doménosztályok

A kezdeti osztálydiagram a játék legfontosabb doménosztályait és azok alapvető kapcsolatait mutatja be. Ezek az osztályok fedik le a játékos karakterét, az akadályok kezelését, a pontozási rendszert, a menürendszert és a hangkezelést. A cél nem az összes technikai részlet felsorolása, hanem a fő logikai elemek azonosítása, amelyekre a későbbi rendszerterv épül.

Az alábbi ábra a főbb osztályokat és kapcsolataikat szemlélteti:



2.1 Doménosztályok rövid leírással

Player – a játékos által irányított karakter, amely folyamatosan fut előre, ugrik, és ütközés esetén game over állapotba kerül. Az osztály tárolja a pozíciót, sebességet, animációs állapotot és az ugrás logikájához szükséges adatokat.

ObstacleManager – az akadályok létrehozását és pályán tartását végzi, gondoskodva arról, hogy a játékos előtt mindig megfelelő kihívás legyen. Időzítés alapján generál új akadályokat, és eltávolítja a képernyőt elhagyó objektumokat.

GameManager – a teljes játékmenet vezérlője, amely koordinálja a frissítési ciklusokat, a játék állapotait (főmenü, játék, szünet, game over), valamint a többi modul közötti kommunikációt. Ő indítja a játékot, kezeli az ütközéseket, és leállítja a futást, amikor a játék véget ér.

ScoreSystem – a pontszám és a high score kezeléséért felelős komponens. Folyamatosan növeli a pontot a megtett távolság és elkerült akadályok alapján, majd a játék végén elmenti vagy betölti a rekordot egy lokális fájlból.

MenuSystem – a főmenü, beállítások menü és Game Over képernyő logikáját valósítja meg. Kezeli a menüpontok kijelölését, a felhasználói választásokat, és jelzi a GameManager felé, mikor kell új játékot indítani vagy kilépni.

SoundManager – a háttérzene és hangeffektek (ugrás, ütközés, menü gombnyomás) lejátszásáért felel. Biztosítja, hogy a hangok a megfelelő pillanatban szólaljanak meg, ezzel erősítve a játékélményt.

2.2 Osztályok közötti kapcsolatok és részrendszerek

A GameManager központi szereplőként kapcsolódik a Player, az ObstacleManager, a ScoreSystem, a MenuSystem és a SoundManager osztályokhoz. Ezek a kapcsolatok jellemzően egyirányú vezérlőkapcsolatok, ahol a GameManager hívja a többi modul metódusait, míg a modulok esemény alapján visszajeleznek a GameManagernek. A Player és az akadályok között ütközés alapú kapcsolat áll fenn, amelyet az ütközésdetektáló logika figyel.

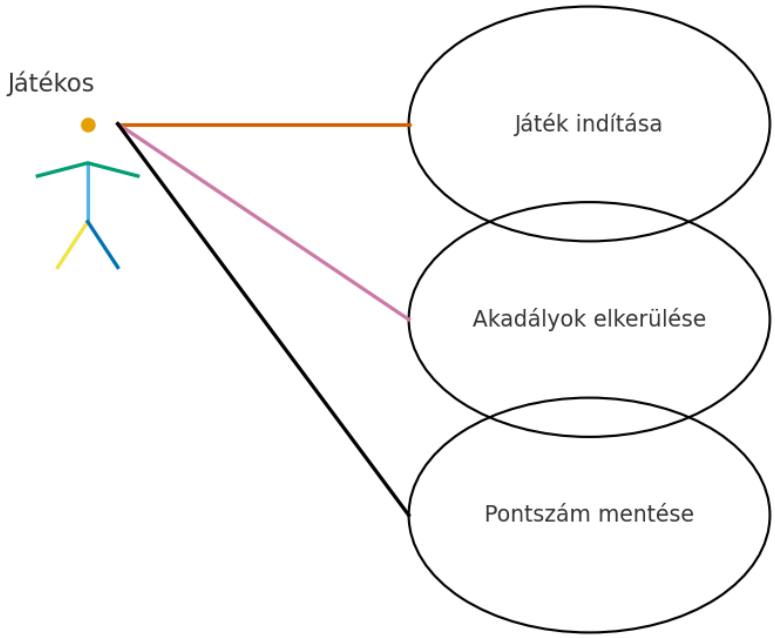
Részrendszerek szempontjából a Player és az akadályok az üzleti logika részét képezik, mivel ők határozzák meg a játékmenetet. A MenuSystem és a UI megjelenítéshez kapcsolódó elemek az UI részrendszerhez tartoznak, míg a ScoreSystem és a fájlból történő olvasás/írás egy egyszerű adattárolási részrendszert alkot. A SoundManager külön alrendszerként értelmezhető, amely a hangkezelésért felel, de szintén szorosan együttműködik a GameManagerrel.

3. Dinamikus modellezés (Use case-ek és szekvenciák)

A dinamikus modell célja, hogy bemutassa, hogyan működnek együtt az egyes objektumok a legfontosabb játékszituációk során. Az Infinite Loopers esetében kiemelt use case-nek tekinthető a játék indítása, az akadályok elkerülése futás közben, valamint a pontszám mentése a játék végén. Ezeket az eseteket a H3-ban meghatározott funkcionális követelmények alapján választjuk ki, és szekvenciadiagramokkal írjuk le a kommunikációt.

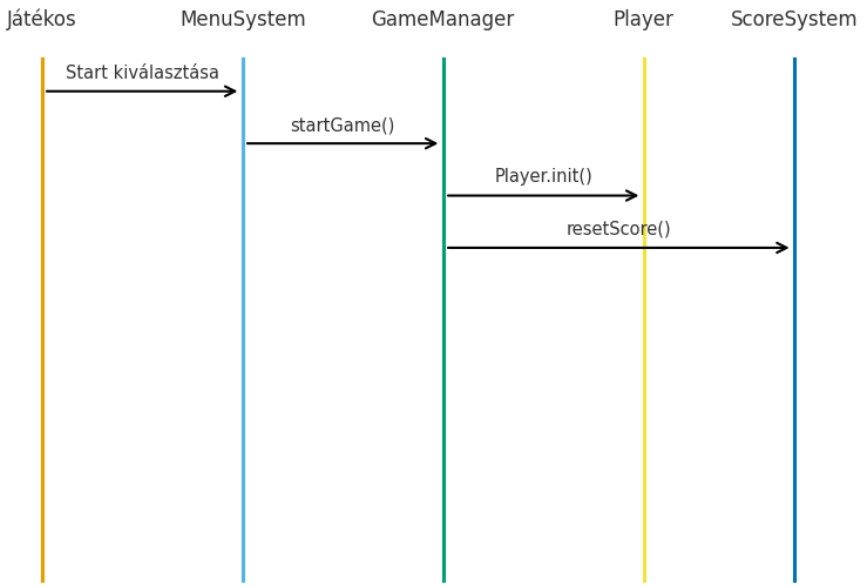
3.1 Use Case diagram

Az alábbi use case diagram a Játékos aktor és a főbb funkciók kapcsolatát szemlélteti. A Játékos a főmenüből indíthatja a játékot, futás közben akadályokat kerül el, majd a játék végén a rendszer elmenti az elért pontszámot.



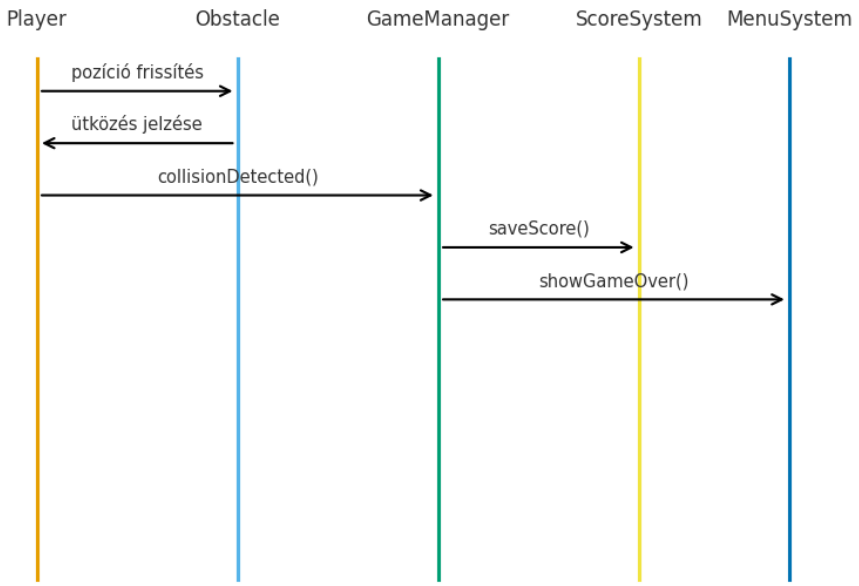
3.2 Szekvenciadiagram – Játék indítása

A következő szekvenciadiagram bemutatja, hogyan indul el egy új játék a főmenüből. A Játékos először a MenuSystemmel lép interakcióba, amely a Start opció kiválasztása után meghívja a GameManager startGame() műveletét. A GameManager létrehozza a Player objektumot, alaphelyzetbe állítja a pontszámot a ScoreSystem segítségével, majd elindítja az akadálygenerálást. Ezzel párhuzamosan a SoundManager elindítja a háttérzenét, és a rendszer áttér a folyamatos játékciklus állapotába.



3.3 Szekvenciadiagram – Ütközés és Game Over

Az ütközés szekvenciadiagramja azt mutatja be, hogyan reagál a rendszer, amikor a Player egy akadállyal ütközik. A Player frissíti a pozícióját, majd az Obstacle objektumokkal történő ütközésvizsgálat eredménye alapján jelzi a GameManager felé, ha ütközés történt. A GameManager ekkor leállítja a játékmenetet, utasítja a ScoreSystemet, hogy mentse el az elért pontszámot, majd értesíti a MenuSystemet, hogy jelenítse meg a Game Over képernyőt. Így a játékos azonnali visszajelzést kap az eredményéről, és dönthet az újrakezdésről.

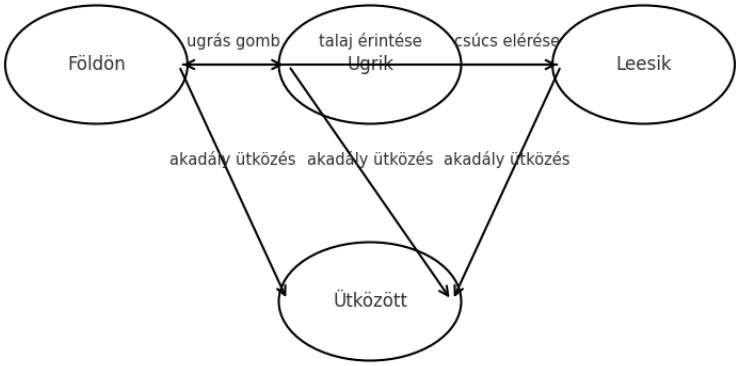


4. Állapotdiagramok

A bonyolultabb objektumok, mint a Player, több különböző állapotban létezhetnek a játékmenet során. Az állapotdiagramok segítenek áttekinteni, hogy egy objektum milyen események hatására lép egyik állapotból a másikba. Az Infinite Loopers esetében a Player állapotai különösen fontosak, mivel a futás, ugrás, esés és ütközés logikája határozza meg a játékélményt és a nehézséget.

4.1 Player állapotdiagram

A Player állapotdiagramja négy fő állapotot tartalmaz: Földön, Ugrik, Leesik és Ütközött. Alapértelmezetten a karakter Földön állapotban van, ahonnan az ugrás gomb lenyomásával Ugrik állapotba vált. Az ugrás csúcspontjának elérését követően a Player Leesik állapotba kerül, majd a talaj elérésekor ismét visszatér a Földön állapotba. Bármelyik mozgás közben bekövetkező akadály-ütközés az Ütközött állapotba viszi a karaktert, amely a Game Over logika aktiválásához kapcsolódik.



5. Finomított osztálydiagram

A dinamikus modellezés után a kezdeti osztálydiagramot pontosítani kell az attribútumok, metódusok és kapcsolatok szintjén. A Player osztály például olyan attribútumokat tartalmaz, mint posX, posY, velocityY és grounded, valamint olyan metódusokat, mint jump(), applyGravity() és update(). Az ObstacleManager egy akadálylista mellett időzítési változókat tárol, és metódusai között szerepel az új akadályok létrehozása és a régiek eltávolítása. A GameManager state attribútuma jelzi a jelenlegi játékmódot (Menu, Playing, Paused, GameOver), míg az update() metódus minden ciklusban meghívja a kapcsolódó modulokat.

A kapcsolatok multiplicitása is pontosításra kerül: egy GameManagerhez pontosan egy Player tartozik, ugyanakkor több Obstacle objektum is jelen lehet egyidejűleg. A ScoreSystem egyetlen példánya kapcsolódik a GameManagerhez, amely a pontszámok frissítéséért felelős. A MenuSystem és a SoundManager szintén egy-egy példányban vannak jelen, mivel a játék során globális szolgáltatásokat nyújtanak. Ez a finomított osztálystruktúra biztosítja, hogy az implementáció átlátható, karbantartható és bővíthető legyen.

6. Összefoglalás

Az analízis modell összefoglalja az Infinite Loopers játék működéséhez szükséges főbb objektumokat, azok kapcsolatait és a játékmenet során lezajló legfontosabb interakciókat. A kezdeti és finomított osztálydiagramok, a use case alapú dinamikus modellek, valamint a Player állapotdiagramja együttesen adják azt a logikai vázat, amelyre a rendszerterv és az implementáció biztonsággal épülhet. A modell kellően részletes ahhoz, hogy támogassa a fejlesztők munkáját, ugyanakkor elég absztrakt marad ahhoz, hogy a későbbi módosítások és bővítések is könnyen beilleszthetők legyenek.