Dokumentáció

Giggly Gug

Készítette:

Jurásek Jónás

Kászonyi Zsombor

Száraz Dániel

Terv:

Az alap elképzelés egy 2D platformer játék volt, amiben néhány pálya található. A játék elkészítéséhez kellő építőelemek:

Menü (Dani):

Alapvető szükséglet egy játékhoz a menü. Mi is beépítettük hát a játékunkba. Van egy alap menü, ami a játék indításakor jelenik meg, itt tudod elindítani a játékot, illetve van egy "in-game" menü, ahol újra tudod indítani a játékot az elejéről, vagy vissza tudsz lépni a menübe. Nyilván lehetőség van a játék folytatására is.

Karakter (Jónás):

A karakter rajzát mi készítettük el, egy az interneten talált kép felhasználásával, amihez testet, illetve lábakat rajzoltunk.

Mozgás (Dani, Zsombi):

A mozgást az alábbi Character Controller-el valósítottuk meg: https://github.com/Brackeys/2D-Character-Controller. Ez adja a mozgás alapjait, ezt egészítettük ki a különböző gombnyomásra történő mozgással.

Karakter animáció (Dani):

Az animációt képekkel valósítottuk meg. Viszonylag kevés (2-3 frame) kép minden animáció, amiket a kezdeti karakter képéből csináltunk. A karakter Animator-jében állítottuk be a feltételeket a különböző animációk közötti váltáshoz. (Pl. ha ugrik, akkor az ugrás animáció játszódik.)

Kamera:

A kamerát úgy akartuk, hogy játékos központú legyen, tehát a játékos mindig a képernyő központjába kerüljön Ehhez a hierarchiában a player alá kellett tenni. Emellett szerettük volna, ha a képernyőn emellett látszódnak az összegyűjtött eszközök, coinok, az eltelt idő, az életek száma. Ehhez egy canvas-t használtunk, aminek átlátszóan hagytuk a hátterét.

Pálya (Jónás):

A pályához egy tilemap-et használtunk, amihez a tile-okat egy letöltött assetből szereztük meg. (https://assetstore.unity.com/packages/2d/environments/nature-pixel-art-base-assets-free-151370) A pályák kialakításánál fő szempont volt, hogy minél változatosabban használjuk a meglévő elemeket, hogy ne legyen egysíkú a játék. Ehhez voltak eszközök az egyes szörnyek, a különböző hátteret díszítő elemek. A pálya kialakításához néhány helyen az egyes mezőket elforgattuk, átméreteztük, hogy ezzel is több eszköz álljon rendelkezésre.

Szörnyek:

A játékban 3 féle szörny található, amiket az asset storeban találtunk. (https://assetstore.unity.com/packages/2d/characters/pixel-monster-pack-75508) A szörnyek között található álló, egyenletesen mozgó, és a játékost követő. A működési elvük, hogy kapnak egy collidert, és amikor a játékos collidere ezzel találkozik, akkor egy script visszarakja a játékost a legutolsó checkpointra.

Álló szörny (Jónás):

A legegyszerűbb szörny, a fent leírtakon kívül semmi mást nem kell hozzárakni.

Mozgó szörny (Dani):

Egyenletesen mozog balra-jobbra, illetve akár lejtőn/ emelkedőn is tud mozogni, amihez a dőlésszögét kell beállítani. Egy hozzá tartozó scripttel mozog, amiben minden frameben egy kicsit odébb rakja a szörnyet, illetve ha elér a kezdőpontjától egy bizonyos távolságot, akkor visszafordul, és visszafelé megy egészen a kezdőpontig. Animáció is van rajta.

Követős szörny (Zsombi):

A követős szörnyhöz 2 dologra volt szükség. - Egy vázra, ami a pályán lévő “nem mozdítható” objektumokat jelzi - Egy szörnyre, ami betartja az említett vázat és nem próbál meg semmin átmenni, hanem kikerüli ezeket.

Az első feladat egy nagyobb falat lett volna, de szerencsére találtunk ehhez már egy megvalósítást így azt könnyen tudtuk használni (https://arongranberg.com/astar/). A második feladat is könnyűnek bizonyult a talált kóddal, azonban némi változtatást vittünk bele, mert máshogy akartuk, hogy működjön. A különbség az volt, hogy ezzel a megoldással amint létrejön a játékos a szörny egyből el kezd felé mozogni. Ez érthető okok miatt nem

volt számunkra ideális, úgyhogy saját követő scriptet kellett írnunk rá. Ehhez persze felhasználtunk néhány beépített függvényt az importált állományból. Ezzel nem csak azt oldottuk meg, hogy ne kezdjen el egyből a játékos felé mozogni a szörny, hanem magába foglalta azt is, hogy több ilyen szörny is szerepelhet egy pályán, hiszen mindegyik egy kör alakú dobozban tud csak mozogni.

Checkpointok (Jónás):

A képe egy asset store-ból szerzett prefab. A célja hogy ne csak a pályák közti átmenetekben mentse el a játékos haladását a játék. Egy colliderjük van, ami ha találkozik a játékos colliderében, akkor a respawn Position-t átállítja a saját pozíciójára.

Háttér (Dani)

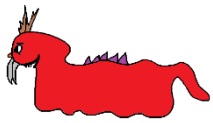
A háttér képet az interneten találtuk meg. A cél az volt, hogy a háttér scrollable legyen, tehát olyan kép kellett, amit ha többször egymás mellé teszünk, akkor nem látszik a képek között az átmenet. 3-szor van a kép egymás mellett, és ha a játékos közeledik az egyik vége felé, akkor egy scripttel átrakódik a másik oldalról az utolsó kép.

Díszítések (Jónás, Dani)

Az asset storeból szedtük a különböző, a pályát díszítő elemeket. Fontos volt, hogy lehessen őket a játékos és a szörnyek elé vagy mögé tenni, ehhez az egyes elemek sorting layerét kell beállítani. Az érmék animációjához a képeket innen szereztük: https://assetstore.unity.com/packages/2d/environments/free-platform-game-assets-85838.

Átlépés a pályák között (Dani):

Rajzoltunk egy lényt, amivel ha érintkezik a játékos, akkor átkerül a következő pályára. Mivel minden pálya egy különböző Scene, ezért ezt úgy valósítottuk meg, hogy az Build Settingsben beállítottuk a Scene-eket, és minden váltásnál az adott Scene után következőt töltjük be a buildIndex segítségével.



Újraéledés:

A halál két módon következhet be:

- A játékos leesik a pályáról
- A játékos egy szörnyhez hozzáér Mindkét esetben azt szerettük volna, ha a játékos mellett az összes szörny az eredeti helyére visszakerül. A játékos visszakerüléséhez szükség volt egy respawnPosition-re, és ha egy bizonyos y érték alá kerül a játékos, vagy hozzáér egy szörnyhez, akkor ide kerül vissza.

Szörnyek újraéledése (Zsombi):

A szörnyek újraéledését a checkpoint-okhoz kötöttük. Amikor a player visszakerül oda, az összes szörny az adott “scene”-en törlésre kerül (amennyiben élt) és egy új kerül a helyükre. Ahhoz, hogy ez működjön a checkpoint-hoz rendeltük hozzá a szörnyeket. Van 2 listája minden checkpointnak az egyik lista x-ik eleme a szörny típusát adja meg, míg a másik lista x-ik eleme a szörny pozícióját tárolja. Miután ezzel rendelkeztünk könnyen létre lehetett hozni a szörnyeket és törölni is könnyen lehetett referencia alapján, mível létrehozáskor egy harmadik listában eltároltuk azt is.

Győzelmi kép (Zsombi):

A győzelmi képnek nem volt alapból ötletünk, de úgygondoltuk, hogy a játék mivel alapvetően vicces elemekből épül fel, ezért megpróbálunk egy mosolyt csalni az ügyes játékos arcára. Az ötlet az volt, hogy kiírja a játékos adatait és a megszerzett pontjait, miközben az egész háttér színesen villog. Ez könnyen megoldható volt, mivel az adatokkal rendelkeztünk, a képernyő villogása pedig csak egy változó állítása (color) a háttér képén.

Pontozás (Dani, Zsombi):

A játékos, ha megnyeri a játékot nem csak az adatait látja, hanem kiíródik a szerzett pontja is. Ezt a következőképpen számoltuk: összegyűjtött érmék * 50 - halálok száma * 20 - eltelt órák száma * 200 - eltelt percek száma * 10/3