STREDNÁ PRIEMYSELNÁ ŠKOLA STROJNÍCKA A ELEKTROTECHNICKÁ

Ulica Fraňa Kráľa 20, 949 01 Nitra

Grumble together

Komplexná odborná práca

Číslo odboru: 2387M



Riešitelia: Timotej Kusy

Adam Kysler

Nitra

2018

Ročník štúdia: štvrtý

Čestné vyhlásenie

Vyhlasujeme, že komplexnú odbornú prácu na tému „*Grumble Together*“ sme vypracovali samostatne, s použitím uvedených literárnych zdrojov. Som si vedomý zákonných dôsledkov, ak v nej uvedené údaje nie sú pravdivé.

|  |
| --- |
| ............................................. |
| Timotej Kusy a Adam Kysler |
|  |

Poďakovanie

Touto cestou by sme sa chceli poďakovať školiteľovi Mgr. Dušan Horváthovi za pomoc, odborné vedenie, cenné rady a pripomienky pri vypracovaní našej práce. Ďakujeme za konzultácie ktoré boli uskutočňované pravidelne, väčšinou dvakrát za mesiac. Tieto konzultácie nám pomohli ďalej napredovať vo vytváraní tejto práce ako aj jej technickej dokumentácie.

**Obsah**

[1. ÚVOD 7](#_Toc507602912)

[2. METODIKA 8](#_Toc507602913)

[2.1 Úvodné informácie 8](#_Toc507602914)

[2.1.1 Použité programy 8](#_Toc507602915)

[2.1.2 Herný engine Unity 8](#_Toc507602916)

[2.1.3 Microsoft Visual Studio 2015 8](#_Toc507602917)

[2.1.4 Ostatné informácie 9](#_Toc507602918)

[2.2 Princíp fungovania hry Grumbling Together 9](#_Toc507602919)

[2.2.1 Pohyb postáv 9](#_Toc507602920)

[2.2.2 Pohyb kamery 11](#_Toc507602921)

[2.2.3 Zaznamenávanie skóre 12](#_Toc507602922)

[2.2.4 Vyhodnocovanie skóre 14](#_Toc507602923)

[2.2.5 Reštartovanie hry (multiplayer) 15](#_Toc507602924)

[2.2.6 Reštartovanie hry (endless) a ukladanie najvyššieho skóre 16](#_Toc507602925)

[2.2.7 Náhodné spawnovanie plošín v endless móde a ich padanie 17](#_Toc507602926)

[2.3 Vytváranie hudby a grafických prvkov hry, animácia a náledné 18](#_Toc507602927)

[2.3.1 Vytváranie hudby pomocou programu FL Studio 18](#_Toc507602928)

[2.3.2 Prostredie programu FL Studio 19](#_Toc507602929)

[2.3.3 Tvorba hlavného menu (grafická časť) 20](#_Toc507602930)

[2.3.4 Tvorba levelov 22](#_Toc507602931)

[2.3.5 Vytváranie grafických prvkov v programe Adobe Photoshop 24](#_Toc507602932)

[2.3.6 Prostredie programu Adobe Photoshop 25](#_Toc507602933)

[3. Výsledky a diskusia 27](#_Toc507602934)

[ZÁVERY PRÁCE 28](#_Toc507602935)

[ZHRNUTIE 29](#_Toc507602936)

[RESUME 30](#_Toc507602937)

[ZOZNAM BIBLIOGRAFICKÝCH ODKAZOV 31](#_Toc507602938)

[ZOZNAM PRÍLOH 32](#_Toc507602939)

**Zoznam obrázkov**

[**1** Prostredie MS Visual Studio (na obrázku kód pre pohyb kamery) 9](#_Toc507525966)

[**2** Použitie kódu zodpovedného za pohyb v inšpektorovi 11](#_Toc507525967)

[**3** Použitie kódu na ovládanie kamery 12](#_Toc507525968)

[**4** Priradenie textového poľa ku kódu 14](#_Toc507525969)

[**5** Kód v inšpektorovi zodpovedný za vyhodnocovanie skóre 15](#_Toc507525970)

[**6**Priradenie funkcie pre reštart hry tlačidlu reagujúcemu na klik 16](#_Toc507525971)

[**7** Vypísanie stavu uloženého skóre v konzole 17](#_Toc507525972)

[**8** Kód v inšpektorovi s priradeným textovým poľom a ďaľsím kódom pre zobrazenie najvyššieho skóre 17](#_Toc507525973)

[**9** Prostredie programu FL Studio 19](#_Toc507525974)

[**10** Klávesy 20](file:///C:\Users\Timo\Desktop\KOP%20.docx#_Toc507525975)

[**11** Tlačidlo a jeho komponenty/funkcie 21](#_Toc507525976)

[**12** Tlačidlo z hlavného menu pre výber levelu 21](#_Toc507525977)

[**13** Prechod po prejdení myšou vo výbere levelu 21](#_Toc507525978)

[**14** Komponenty Box Collider 2D a Rigidbody 2D 22](#_Toc507525979)

[**15** Model sústruhu 23](file:///C:\Users\Timo\Desktop\KOP%20.docx#_Toc507525980)

[**16** Herný objekt - robot ABB IRB120 23](file:///C:\Users\Timo\Desktop\KOP%20.docx#_Toc507525981)

[**17** Sprite editor - postava určená na animáciu 24](#_Toc507525982)

[**18** Prostredie programu Adobe Photoshop 25](#_Toc507525983)

# ÚVOD

Herný priemysel je v posledných rokoch na vzostupe a teší sa veľkej popularite najmä medzi tínedžermi, ale aj starší ľudia si v ňom nájdu to svoje. Možnosti vývoja hier sú prakticky neobmedzené a existuje mnoho žánrov hier ako napríklad Adventúry, Akčné, RPG či Arkády.

V Adventúrach prevláda hlavne dejová línia a hráč je plne ponorený do psychologického vývoja postáv a častokrát rozhoduje o tom, čo sa s postavou ďalej stane a ako pokročí dej. Dnes sú veľmi populárne a prejdenie dejovej línie môže trvať aj 100 hodín.

Arkády sú 2D hry, niekedy nazývané aj „plošinovky“ a boli v obľube hlavne v 90.tych rokoch. Boli to prvé hry ktoré sa objavili na obrazovkách moderných počítačov. V hre ide napríklad o dosiahnutie čo najvyššieho skóre, alebo predbehnutie súpera alebo dosiahnutie nejakého cieľa. Najznámejšími hrami boli napríklad séria Mario Bros alebo Sonic. S odstupom času boli arkádové hry vytlačené z popredia inými žánrami a stali sa tak akousi nostalgickou záležitosťou, avšak neupadli do zabudnutia a vedia byť pútavé aj dnes a sú stále vyvíjané.

Akčné hry hľadajú zmysel v eliminácii oponenta hrubou silou. Zlepšujú senzorické funkcie a pozornosť. Najznámejšou hrou je asi Counter-Strike od spoločnosti Valve. Akčné hry sú uznané ako šport a majú svoje majstrovstvá sveta.

RPG hry sú masívne hry ktoré v sebe skĺbujú príbeh, akciu a vývoj postáv. Príbeh je veľmi prepracovaný a má veľa odbočiek. Hráč si postavu vylepšuje za získané body skúseností zo súbojov alebo úloh. Medzi najznámejšie RPG hry patrí séria The Elder Scrolls.

Pre našu prácu sme si zvolili práve arkádovú hru pre jej zábavnosť, schopnosť stimulovať senzorické funkcie nášho tela, rozvíjajú súťaživosť a hravosť a z vyššie spomínanej nostalgie.

Témou našej práce je vytvorenie 2D „plošinovky“ pre dvoch hráčov kde sa obaja hráči budú snažiť predbehnúť v nazbieranom skóre a tak v sebe rozvíjať súťaživosť. Hra je zasadená do prostredia školy, presnejšie do troch vyučovaných odborov na našej škole. Každý level je jeden odbor. Prvý level je odbor elektrotechnika, druhý je strojárstvo a tretí je mechatronika. Posledný level je vytvorený v štýle „endless“ a teda nekonečný. Hráč zbiera skóre po generovaných platformách, ak zomrie jeho skóre sa uchová a hrá odznova.

Na vytvorenie hry a spojenie všetkých aspektov hry sme použili softvér Unity spolu s Microsoft Visual Studio ako IDE (Integrated Development Environment – vývojové prostredie), Adobe Photoshop pre tvorbu grafiky a FL Studio pre tvorbu hudby. Hra má odrážať naše technické zručnosti v oblasti programovania, vytvárania grafických prvkov a ovládania profesionálneho softvéru.

# METODIKA

# Úvodné informácie

# Použité programy

Pri vytváraní tejto práce sme využívali Microsoft Visual Studio 2015 pre programovanie v programovacom jazyku C# (C sharp) s použitím knižníc Unity API, pre vytváranie všetkých herných textúr program Adobe Photoshop, pre vytváranie hudby FL Studio a na následné spojenie hry herný engine Unity.

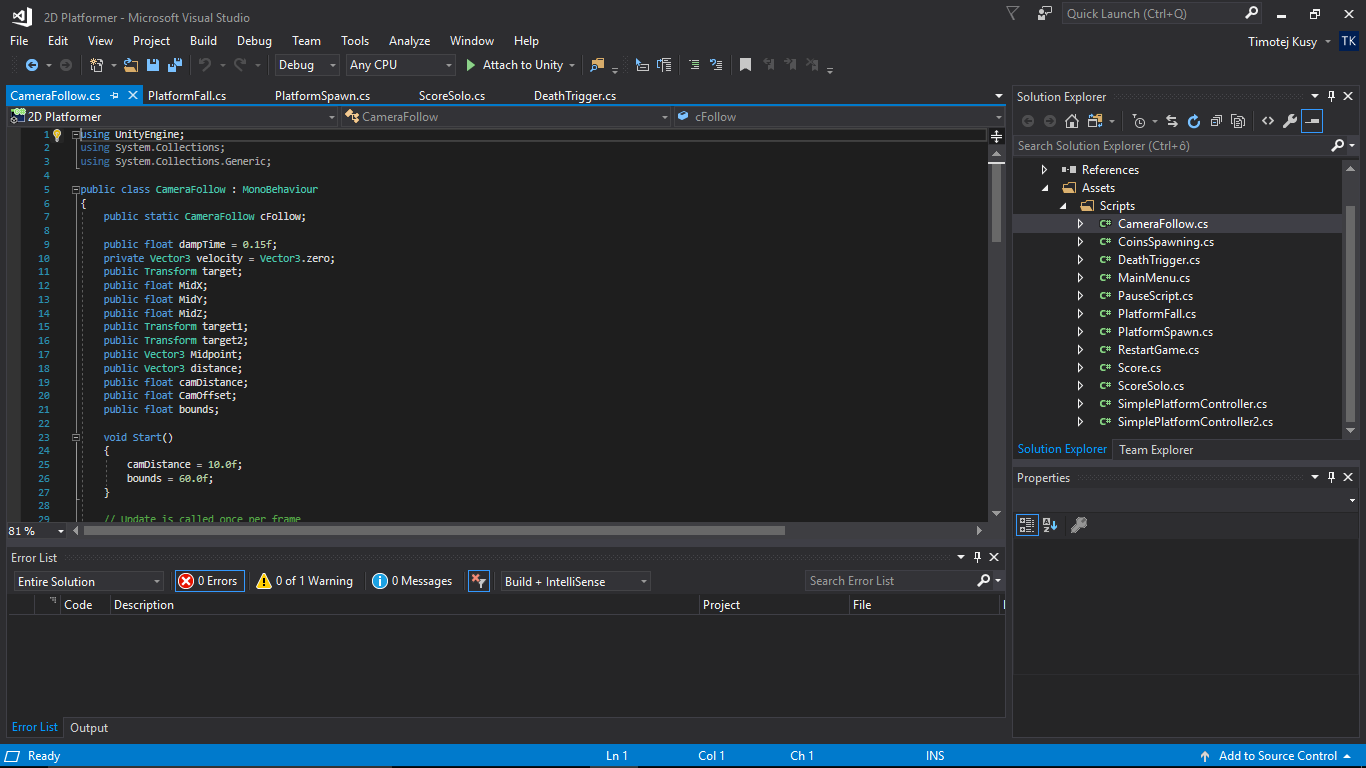
# Herný engine Unity

Unity je multiplatformový herný engine vytvorený spoločnosťou Unity Technologies v roku 2005. Unity je viacúčelový a ponúka možnosť vytvárať hry v 3D ale aj v 2D zobrazení. Unity podporuje kompiláciu hier pre 27 platform. Použili sme verziu Personal lebo je zadarmo pokiaľ náš ročný obrat nepresiahne 100 000 dolárov.

# Microsoft Visual Studio 2015

Microsoft Visual Studio je vývojové prostredie od spoločnosti Microsoft a primárne bolo vytvorené pre jazyk C# (C sharp) keďže ide o jazyk C, upravený práve spoločnosťou Microsoft. Visual Studio sme si zvolili práve pre jeho výbornú spoluprácu s Unity. Navyše ponúka automatické dopĺňanie či už údajových typov, tried alebo opravu chýb. Táto funkcia veľmi pomáha zefektívniť prácu. Ďalej Microsoft nezabudol na naše oči a keďže biela farba je celkovo silná a ťahá oči, používali sme tmavý motív.

Používali sme verziu Community lebo je zadarmo pre kohokoľvek. Samozrejme sú na výber aj platené verzie pre profesionálov ktoré sa predplácajú ročne. Na naše potreby bohato stačila verzia Community. Napríklad verzia Professional stojí 1199 dolárov na rok a verzia Enterprise stojí 5999 dolárov na rok.



**1** Prostredie MS Visual Studio (na obrázku kód pre pohyb kamery)

# Ostatné informácie

Táto hra vznikla medzi májom 2017 a marcom 2018 so zámerom vytvoriť hru na základe našich nadobudnutých schopností v oblasti programovania a vytvárania grafického obsahu ako aj prácu s profesionálnym a rozsiahlym softvérom. Hra je v anglickom jazyku a aj názvy jej súčastí z dôvodu že všetok použitý softvér je v anglickom jazyku, pomohlo nám to v prehľadnosti a jasnosti našej práce. Miešanie dvoch jazykov by bolo mätúce a preto aj názvy premenných sú písané anglickým jazykom pre naše osobné potreby.

# Princíp fungovania hry Grumbling Together

V tejto kapitole si zhrnieme a vysvetlíme princípy fungovania hry a jej komponentov. Budeme sa venovať najdôležitejším častiam. Kvôli obmedzenému rozsahu dokumentácie menej dôležité súčasti vynecháme ale okrajovo ich spomenieme aby sme doplnili súvislosti.

# Pohyb postáv

Pohyb postáv je zabezpečovaný zdrojovým kódom SimplePlatformController. Hráči pohybujú postavami vľavo, vpravo a hore, teda v osiach x a y. Každá postava musí obsahovať komponent „Rigidbody2D“ ktorý zabezpečuje pohyb tak, že prepíše komponent „Transform“ čo je vlastne komponent zobrazujúci polohu v súradnicovom systéme.

Kód si zo vstupu „Horizontal“ zistí horizontálnu os a priradí ju premennej „h“, potom zistenú hodnotu zapísanú do „h“ priradí ovládaču animácie ako parameter pre rýchlosť, hodnota musí byť priradená v absolútnom stave aby nebola záporná a tým pádom ak sa budú hráči pohybovať doľava a doprava vždy bude mať rýchlosť kladnú hodnotu.

Ďalšia podmienka zisťuje či rýchlosť postáv je nižšia ako nami zadaný limit a ak je nižšia, tak pridá silu pohybu ktorá je uložená do premennej „moveForce“, premenná „h“ pridá parameter smeru sily. Ak h= -1 postavy pôjdu doľava ak h= 1 postavy pôjdu doprava.

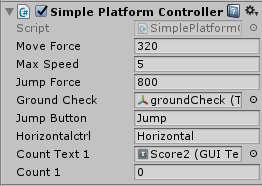
Pre zabezpečenie konštantnej rýchlosti podmienka zisťuje či sme neprekročili nami zadanú maximálnu rýchlosť. Ak zistí prekročenie limitu, rýchlosť vráti na nami zadanú hodnotu a tak isto je pohyb zabezpečený práve len pre ten smer ktorým chceme ísť.

Pohyb doľava a pohyb doprava sú pre prvú postavu klávesy A a D a pre druhú postavu sú to pohybové šípky doprava a doľava. Klávesy sú priradené mimo kódu a to v prostredí Unity, konkrétne v správcovi vstupov. Pre vstup Horizontal je negatívna hodnota a teda vľavo, nastavená na klávesu „a“ a pozitívna na „b“.

Skákanie je riešené tak, že kód si každú snímku zistí či je postava postavená na zemi. V deklarácii je premenná „grounded“ nastavená ako false. Ak podmienka zistí že herný objekt typu „Physics2D“ pomenovaný v Unity ako „grounded“ sa dotýka hladiny „Ground“ teda zemi, a bude stlačené tlačidlo skákania, nastaví premennú „jump“ na true, zapne sa animácia skákania, nami deklarovaná sila skoku sa prenesie do hry a výsledkom bude to že postava vyskočí. Či je postava na zemi sa zisťuje pomocou premennej „groundCheck“. Keďže je triedy Transform a je verejná, môžeme v hernom inšpektorovi v Unity pridať herný objekt „groundCheck“ ktorý je na spodnej hrane postavy. Po skončení animácie sa hráč vráti naspäť na zem a „jump“ sa znovu nastaví na false. Klávesy pre skok sú priradené v Unity, presnejšie v správcovi vstupov pre vstup Jump ktorý platí pre os x.

V inicializácii sú postavy otočené smerom doprava. Je to preto, lebo sú tak nakreslené. V kóde je to tiež deklarované ako true. Funkcia „Flip“ teda otočenie, funguje tak že ak premenná „facingRight“ typu bool je false, na komponent transform sa na parameter osi x prenesie -1 a tým pádom sa hráč otočí na osi x doľava. Ak je premenná „h“ väčšia ako 0, to znamená že hráč smeruje doprava, ale nie je otočený doprava vyvolá sa funkcia „Flip“ inak ak je „h“ menšie ako 0, to znamená že postava smeruje doľava, a postava je otočená doprava, tiež vyvolá funkciu „Flip“.

Na obrázku nižšie je možno vidieť časť herného inšpektora s priradeným kódom na pohyb hráča. Ako možno vidieť, ku Ground Check je priradený „Transform“ komponent groundCheck hráča. K Jump Button (skákacia klávesa) je priradená značka Jump z inšpektora vstupov a pre horizontálny pohyb je priradená značka Horizontal. Pre hráča 2 je priradená značka Jump2 a Horizontal2.



**2** Použitie kódu zodpovedného za pohyb v inšpektorovi

# Pohyb kamery

Kamera je základom každej hry. Vykresľuje pohľad na hernú scénu. Môže sa vzďaľovať, približovať alebo prenasledovať herné objekty. My sme využili poslednú možnosť, a teda prenasledovanie postáv. Ovládanie kamery zabezpečuje zdrojový kód CameraFollow.

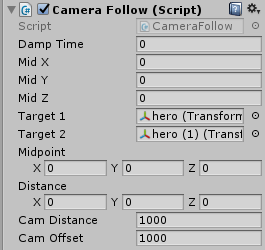
Začiatkom kódu je funkcia „Start“. Táto funkcia je vyvolaná vždy na začiatku spustenia hry. V tejto funkciu sme si nastavili odstup kamery. Odstup kamery je nastavený na hodnotu 10.

Funkcia „Update“ je vyvolávaná každou aktualizáciou snímky a zabezpečuje nám všetok pohyb kamery. Kód sleduje polohu oboch postáv. Deklarovali sme si verejné premenné „Target1“ a „Target2“ triedy „Transform“. Táto trieda definuje polohu a rotáciu herného objektu. Každá postava má komponent transform. To že sú verejné znamená, že v inšpektorovi v Unity k nim priradíme postavy a tak kód bude získavať ich súradnice. Premennú „distance“ (vzdialenosť) sme deklarovali rozdielom polôh postáv.

Podmienka robí to, že ak je vzdialenosť kamery menšia alebo rovná tak ju nastaví na hodnotu 10. To nám zabezpečí stálu vzdialenosť kamery od hernej plochy. Ďalej ďalšia podmienka zabezpečuje odstup kamery. Odstup kamery je daný vzdialenosťou medzi hráčmi na osi x ktorá je vynásobená číslom 0.3. Ak by sme tak nespravili a postava sa by sa vzdialila od druhej tak by kamera nesledovala ani jednu postavu a obe by boli mimo obrazovky. Takto sa postava zastaví a nepustí ju ďalej ak je príliš vzdialená od druhej.

Kamera získava súradnice kam sa má dostať od samotných postáv. Pozícia postáv na osi x je zapísaná do premennej „MidX“ a pozícia postáv na osi y je zapísaná do „MidY“ a obe sú vydelené ešte dvomi. Tieto súradnice nám určujú stredný bod „Midpoint“.

Deklarovali sme si novú premennú „point“ štruktúry „Vector 3“. Premenná získa komponent „Camera“ z Unity ktorá získa prístup k nášmu midpointu a prekonvertuje ho z kódu do Unity. Vytvorili sme si premennú „delta“ a tú definovali tak, že sme od stredného bodu odčítali kameru ktorá získala hodnoty z odstupu kamery a vzdialenosti kamery a následne sme kamere určili destináciu tak že sme k jej pozícii pripočítali „delta“. Jej pohyb je teda zabezpečený zistením polohy hráčov a prenesenie ich hodnôt na seba + čas oneskorenia lebo inak by sa všetko dialo príliš rýchlo. Na obrázku nižšie môžete vidieť časť herného inšpektora kde je použitý kód na ovládanie kamery. Ako možno vidieť, Target 1 a Target 2 majú priradené „Transform“ komponenty hráčov.



**3** Použitie kódu na ovládanie kamery

# Zaznamenávanie skóre

Obe postavy a teda obaja hráči majú svoj vlastný ukazovateľ skóre dosiahnutého zbieraním mincí. Tento ukazovateľ sme v Unity pridali pomocou „Canvas“ – plátna. Na plátno sa ukladajú UI komponenty ako je textové pole a tlačidlo. Textové pole má označenie „GUIText“.

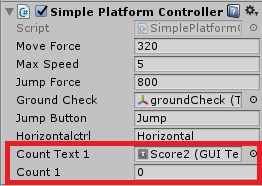
Zaznamenávanie kolízií pre priradenie skóre je riešené komponentmi herných objektov triedy „Collider 2D“ a je implementované do zdrojového kódu pohybu. „Collider 2D“ je komponent vďaka ktorému vieme zaznamenať kolíziu herných objektov a zároveň je triedou pre naše premenné v kóde.

Kód začína deklarovaním si premennej „countText1“ (count, z ang. počítať) triedy herného objektu Unity, GUIText. Táto premenná bude prenášať hodnoty dosiahnutého skóre do samotnej hry, konkrétne bude skóre zobrazovať na obrazovke s použitím príslušnej funkcie. Ďalej sme si deklarovali premennú „count1“ údajového typu int. Táto premenná nám bude uchovávať dosiahnuté skóre v hre pre neskoršie použitie. Vo funkcii „Start“ sme si „count1“ nastavili na 0 a vyvolali funkciu „SetCountText“ ktorá nám zobrazí skóre na obrazovke v samotnej hre. Toto nám zabezpečí že vždy pri spustení hry bude skóre vynulované. Pretože je to zapísané vo funkcii „Start“, tento príkaz sa vykoná len keď bude hra spustená a nikdy inokedy a to platí aj o vyvolaní funkcie.

Vo funkcii „SetCountText“ použijeme našu premennú typu GUIText a „count1“. Výsledok ešte prekonvertujeme do údajového typu String. Ak by sme tak nespravili Unity by nevedelo zobraziť výsledok v textovom poli na „plátne“.

Kód získava prístup k textovému poľu prostredníctvom herného inšpektora v Unity. Keďže naša premenná „countText1“ je verejná, môžeme ju vidieť v inšpektorovi v Unity a priradiť mu textové pole. To dosiahneme jednoduchým presunutím textového poľa do inšpektora.

Ak sa hráčov „collider“ dotkne „collidera“ iného herného objektu, pripočíta sa skóre a iný herný objekt, v našom prípade minca, sa deaktivuje a vyvolá sa funkcia „SetCountText“ ktorej prácu sme si vysvetlili vyššie. Ako ale hra bude vedieť že sa hráč dotkol práve mince a nezapočíta skóre pri dotknutí napríklad s iným hráčom alebo plošinou či iným herným objektom? V inšpektorovi Unity sme zbieraným objektom, minciam, dali „tag“ (značka) „PickUp“ (z ang. zodvihnúť). Pridaním podmienky vo funkcii „OnTriggerEnter2D“ sme zabezpečili to, že skóre sa pripočíta práve vtedy ak sa hráč dotkne herného objektu s označením „PickUp“. Na obrázku nižšie je možno vidieť časť herného inšpektora s priradeným kódom na pohyb hráča ale dôležitejšie je v tejto časti skóre. Ku Count Tex 1 je priradený GUI Text Score.



**4** Priradenie textového poľa ku kódu

# Vyhodnocovanie skóre

Vyhodnocovanie skóre je dôležitá súčasť hry nakoľko ide o hru ktorá je pre dvoch hráčov. Na konci kola hra vyhodnotí ktorý hráč mal vyššie skóre a vypíše ho na obrazovku. Toto zabezpečuje kód „Score“ ktorý si nižšie popíšeme.

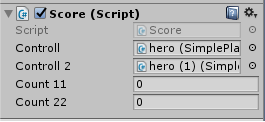
Deklarovali sme si triedy „SimplePlatformController/2“ a nazvali sme ich „controll“ a „controll2“. Tieto triedy nám umožňujú získavať informácie mimo aktuálneho kódu, v tejto situácii z kódov pre ovládanie. Ďalej sme si deklarovali premenné „count11“ a „count22“ údajového typu int ktoré budu uchovávať skóre hráčov pre vyhodnotenie.

Vo funkcii Update ktorá je vyvolávaná každú snímku priraďujeme premenným count11 a count22 hodnoty z premenných count1 a count2 z pohybových kódov tak, že im povolíme prístup do pohybových kódov keďže sú verejné a tak isto aj premenné v nich sú verejné.

Vo funkcii „OnTriggerEnter2D“ máme podmienku ktorá zabezpečuje to že ak sa hráč dotkne herného objektu „Finish“, teda cieľu, tak sa cieľ deaktivuje a vyvolá sa funkcia „GameOver“ ktorá ukončí hru.

Funkcia „GameOver“ porovná skóre pomocou podmienok. Ak je skóre 1 hráča 1 vyššie ako skóre druhého hráča nahrá novú scénu kde bude vypísané že vyhral hráč 1. Ak bude skóre 2 vyššie ako skóre hráča 1, načíta sa scéna kde bude vypísané že vyhral hráč 2. Odtiaľto môžu ísť hráči do menu a vybrať si znovu level aký chcú.

Na obrázku nižšie možno vidieť časť herného inšpektora, konkrétne časť „Finish“, kde ku Controll a Controll 2 sú priradený hráči – hero a hero (1). Unity si už na základe kódu z nich zoberie to potrebné a teda získa prístup k uchovanému skóre. Count11 a Count22 sme zvolili verejné pre možnosť real-time sledovania hodnôt. Ak by boli privátne tak by sme túto možnosť nemali.



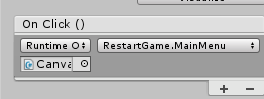
**5** Kód v inšpektorovi zodpovedný za vyhodnocovanie skóre

# Reštartovanie hry (multiplayer)

Reštartovanie hry v multiplayer móde je riešené cez „PlayerPrefs“. Táto trieda uchováva a získava prístup k preferenciám hráča medzi hernými scénami. Ak by sme túto triedu nepoužili, nadobudnuté hodnoty by sa stratili pri prepnutí scény.

V kóde „Score“ sme do preferencíí hráča zapísali číslo scény v ktorej sa práve nachádzame. Prečo v kóde „Score“? No preto lebo je to kód na cieľovej lajne a nezabezpečuje nám len vyhodnocovanie skóre ale tak isto aj ukončenie hry. Ak sa hra reštartuje tak číslo scény zostane uchované pre budúce použitie. Pri načítaní výhernej scény sa spustí aj kód „RestartGame“. Tento kód následne získa názov posledného levelu a tým pádom nám umožní hru reštartovať. Ďalšou možnosťou je návrat do hlavného menu teda do scény „0“.

Na výhernej obrazovke sú dve tlačidlá. Jedno je pre návrat do hlavného menu a druhé pre reštartovanie hry. Tieto tlačidlá musia byť uložené na „plátne“ inak by sme nemohli získať prístup ku kódu a nemohli by sme nimi ovládať hru. V hernom inšpektorovi sme priradili zdrojový kód na „plátno“ a tlačidlá urobili jeho „deťmi“. V komponente „Button (Script)“, teda ovládač tlačidla pomocou kódu, sme priradili ako prístupový objekt „plátno“ a v sekcii pre reakciu na klik sme priradili funkciu z kódu zodpovednú či už za návrat do menu alebo reštartovanie levelu.



**6**Priradenie funkcie pre reštart hry tlačidlu reagujúcemu na klik

# Reštartovanie hry (endless) a ukladanie najvyššieho skóre

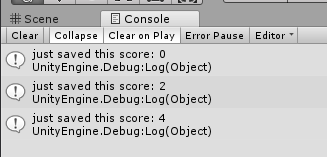
Reštartovanie hry v endless móde je trochu odlišné. Museli sme vyriešiť problematiku uchovávania najvyššieho skóre aj po znovunačítaní scény.

Reštartovanie hry prebehne iba ak hráč spadne z plošiny a dotkne sa tzv. „DeathTrigger“ ktorý indikuje hráčovu smrť a vyvolá funkciu pre reštartovanie hry. Funkcia pre reštartovanie hry funguje na princípe kolízií a tvorí ju podmienka. Podmienka hovorí že ak sa herný objekt so značkou „Player“ dotkne nášho „DeathTrigger-u“ znovu načíta scénu.

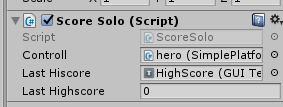
Tu sa dostávame k problému so spracovaním a zobrazením najvyššieho skóre. Po znovunačítaní scény sa skóre vymaže. Preto sme toto ukladanie skóre museli riešiť cez Preferencie hráča, podobne ako pri reštartovaní hry pre viac hráčov. V kóde sme si deklarovali viacero premenných. Najprv sme museli získať prístup ku skóre v reálnom čase. To sme vyriešili deklarovaním premennej „controll“ triedy „SimplePlatformController“, ktorá zároveň odkazuje na kód kde sa skóre započítava. Táto premenná je verejná a preto sme k nej v hernom inšpektorovi priradili hráča a kód tak získal prístup ku kódu „SimplePlatformController“. Vytvorením privátnych premenných typu int, „count11“ a „hiScoreCount“ sme získali prístup k aktuálnemu skóre, ktoré sme vo funkcii „Update“ priradili k premennej „count11“.

Vo funkcii „FixedUpdate“ je podmienka ktorá zabezpečuje že sa skóre uloží a vypíše. Hovorí že ak je aktuálne skóre väčšie ako najvyššie tak najvyššie skóre je rovné aktuálnemu. Potom do preferencií hráča prenesieme skóre pod názvom „HighScore“ a následne uložíme. Vo funkcii „Awake“ použijeme premennú „lastHighScore“ údajového typu int a priradíme jej naše uložené skóre. Tento príkaz musí byť len v tejto funkcii keďže je vyvolaná len raz za celú dobu hry a to práve pri spustení. Následne toto skóre vypíšeme na obrazovku pomocou textového komponentu v Unity ale musíme hodnotu skóre prekonvertovať na údajový typ String lebo sme ho ukladali ako integer a Unity by ho nevedelo vypísať.

Zdrojový kód je priradený k hernému objektu „ScoreManager“. Pre kontrolu funkčnosti kódu a ukladania skóre sme pridali ešte príkaz na vypísanie do konzoly, konkrétne to že uložilo skóre a jeho hodnotu.



**7** Vypísanie stavu uloženého skóre v konzole



**8** Kód v inšpektorovi s priradeným textovým poľom a ďaľsím kódom pre zobrazenie najvyššieho skóre

# Náhodné spawnovanie plošín v endless móde a ich padanie

Náhodné spawnovanie plošín je závislé od polohy hráča a od randomizácie na úrovni kódu. Princíp spočíva v tom, že ak sa kamera dostane do určitej polohy a nebude tam plošina tak ju tam vytvorí. Plošiny sa náhodne spawnujú v osi x a aj v osi y, čo znamená, že majú medzi sebou rozdielnu vzdialenosť aj výšku a preto sa táto hra stáva pre hráča ťažšia. Navyše hráč musí plošinu čo najrýchlejšie opustiť inak spadne spolu s ňou.

Najskôr kód zisťuje polohu plošiny v osi x a y. Následne tieto hodnoty priradí k premenným vytvoreným pre uchovávanie dĺžky a výšky. V Unity máme vytvorený herný objekt nazvaný „PlatformGenPoint“, teda bod, ktorý je bodom pre generáciu pre novú plošinu. Tento bod sa hýbe s kamerou a ak podmienka v kóde zistí že poloha tohto bodu je menšia ako poloha „SpawnManagera“ , tak prejde na príkazy ktoré náhodne vyhodnocujú vzdialenosť medzi plošinami v osiach x a y. Tieto vzdialenosti sú ale obmedzené nami zadanými minimálnymi a maximálnymi hodnotami vzdialeností a preto kód náhodne vyhodnocuje len v nami stanovenom rozpätí.

Ďalší príkaz ktorý podmienka vykoná je vytvorenie novú kópiu plošiny a to tak, že nastaví jej novú polohu s hodnotami ktoré sme zistili skôr.

Aby kód fungoval museli sme mu priradiť herné objekty. Aby vedel čo má spawnovať tak sme mu v hernom inšpektorovi priradili platformu ktorej kópiu má vytvoriť spolu aj s mincami. Tak isto sme mu museli priradiť bod pre generáciu nových plošín aby kód vedel zistiť jeho polohu a vykonať vyššie opísané príkazy.

Pre zvýšenie náročnosti hry sme sa rozhodli použiť kód ktorý zabezpečuje padanie plošín. Ak hráč stupí na plošinu, začne sa odpočet a po uplynutí stanoveného času plošina spadne a ak hráč nestihne preskočiť na inú, tak spadne aj on a musí hrať odznova. Odpočet sme nastavili na 0.35. Je to dostatočný čas na preskočenie z jednej plošiny na druhú ale môže byť obtiažne získať všetky mince vygenerované na plošine a tak hráč musí reagovať rýchlo.

V kóde to zabezpečuje podmienka ktorá hovorí, že ak sa hráč dotkne iného herného objektu, v tomto prípade plošiny, tak sa vyvolá funkcia „Fall“ ale s oneskorením ktoré sme v deklarácii síce stanovili ale keďže je verejné tak ho môžeme meniť aj v Unity podľa potrieb. Funkcia „Fall“ nastaví „isKinematic“ na false. To znamená že komponent „RigidBody2D“ začne reagovať na kolízie a sily pôsobiace naň a spadne z obrazovky.

Kódy sme sa snažili spraviť čo najefektívnejšie. V začiatkoch obsahovali veľmi veľa príkazov a postupom času sme ich zjednodušili. Pri hre takéhoto typu sme nepotrebovali odlaďovať využitie procesoru, pamäte RAM a grafickej karty nakoľko hra je, ako sme vyššie spomínali, efektívna.

# Vytváranie hudby a grafických prvkov hry, animácia a náledné

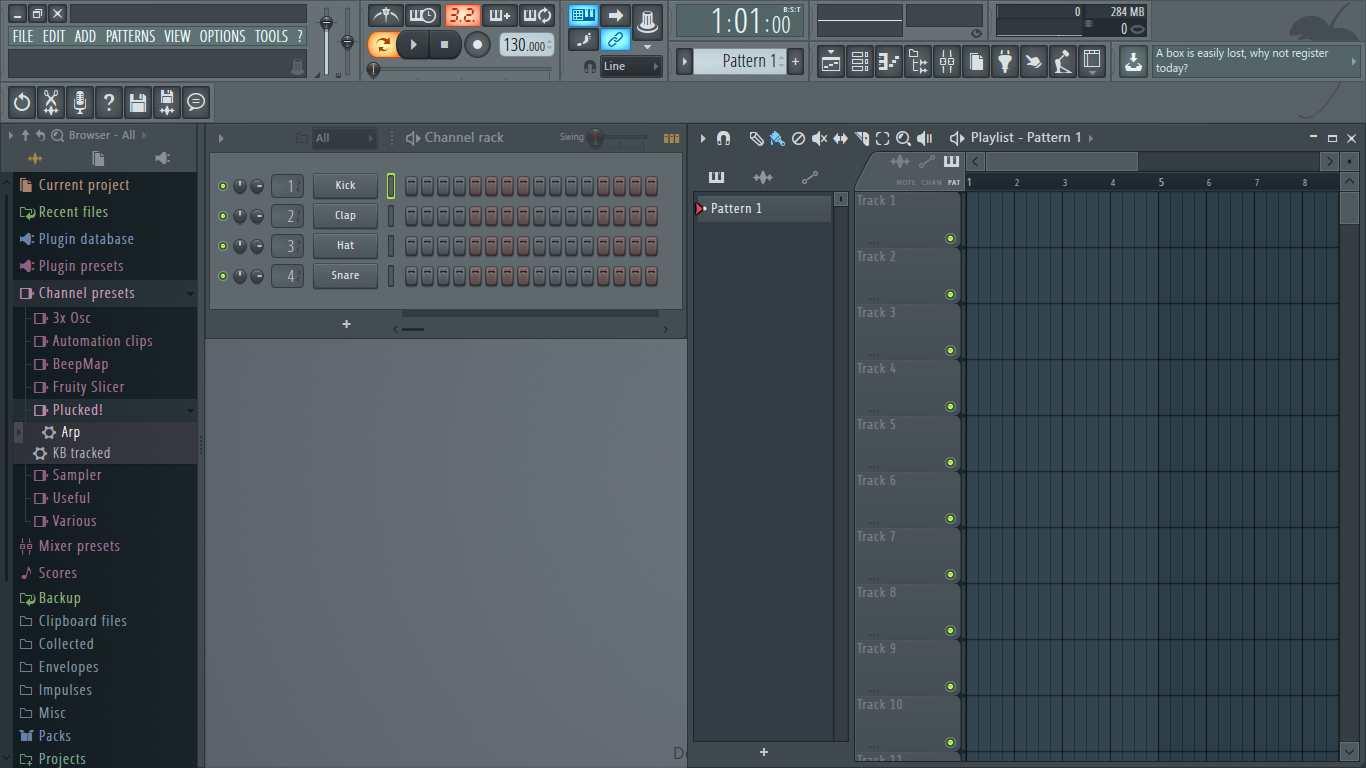
V tejto kapitole si vysvetlíme ako sme vytvárali hudbu a grafické prvky do našej hry. Na tvorbu grafických prvkov sme použili softvér Adobe Photoshop a na tvorbu hudby FL Studio 12. Celé sme to spojili v Unity.

# Vytváranie hudby pomocou programu FL Studio

# 

FL Studio 12 je program od belgickej spoločnosti Image-Line a slúži na úpravu a tvorbu hudby. My sme používali demo verziu, ktorá ponúka v podstate všetky pluginy no po uložení projektu nie je možné úplne otvoriť projekt znovu. Nedá sa teda pokračovať v rozrobenej práci. Z pluginov sme používali najmä tie základné priamo od tvorcov. Z dôvodu slabších skúseností s programom sme sa snažili komponovať jednoduchšie melódie, ktoré sa aj hodia do hry ako je tá naša.

# Prostredie programu FL Studio



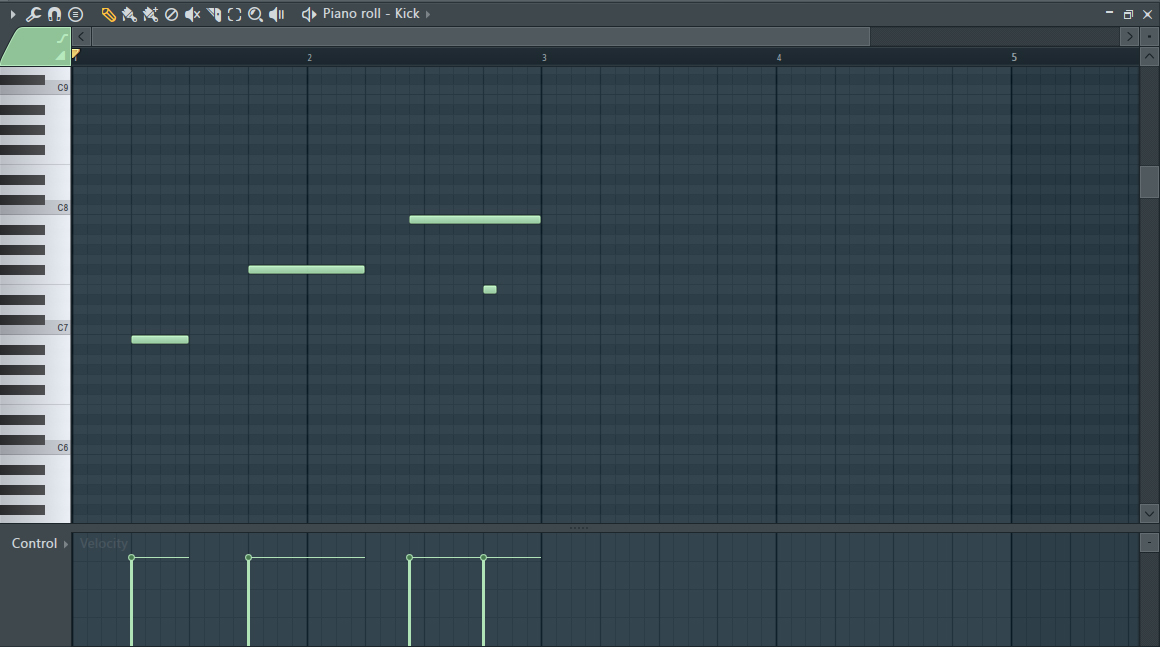
**9** Prostredie programu FL Studio

Vľavo hore sa nachádzajú nastavenia súborov, nastavenia vzorov a iné možnosti. Pod nimi sa nachádza vyhľadávač pluginov, čo sú vlastne rozšírenia. Napravo od neho sú nastavenia kanálov, cez ktoré sme aj pridávali rôzne zvuky do vzorov, ktoré sa nachádzajú hneď napravo. Vzory sa dajú priamo presúvať vedľa na časovú os kde sa už všetko spája a upravuje.

Do ovládača kanálov môžeme pridávať ďalšie kanály pomocou pluska v dolnej časti okna. Zvuky potom môžeme pridávať buď do interaktívneho krokového triediča (interactive step sequencer) alebo pomocou kláves (piano roll). Jednotlivé kanály majú ešte širšie nastavenie a ich výsledný tón sa dá editovať.

Vzory sa neskôr presunú na časovú os, na ktorej už vzniká hotová skladba, ktorá sa dá samozrejme ešte editovať. Ak máme už na osi niečo vytvorené a zmeníme jeden vzor, zmena sa prejaví hneď a v celom projekte.

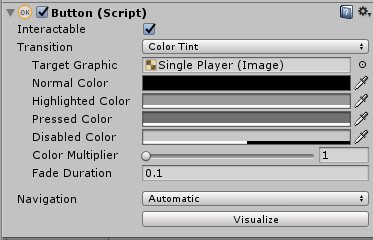
Širšia variácia tónov sa dá dosianuť práve pomocou kláves. Dá sa tu zvoliť presná výška tónu a aj dĺžka. Dajú sa kombinovať rôzne výšky. V hornej časti sa nachádzajú nástroje ako napríklad kreslenie (práve aktívna ceruzka), mazanie, selekcia atď. Takto vytvorené vzory sa dajú prehrať (nad číslom 2 na časovej osi). Časová os je rozdelená na sekundy. Je dobré robiť niekoľko sekundové vzory, ktoré sa neskôr kombinujú. Dole ešte vieme ovládať rýchlosť a hlasitosť jednotlivých úsekov vzoru.



**10** Klávesy

# Tvorba hlavného menu (grafická časť)

Menu sme v Unity robili pomocou Tlačidiel. Tlačidlá sú tvorené z textu a pozadia, ktoré sa mení po ukázaní kurzoru na tlačidlo a potom aj po kliknutí. Interakcia tlačidla sa dá meniť v nastavení tlačidla.



**11** Tlačidlo a jeho komponenty/funkcie

Prechody je možné meniť medzi sfarbením, výmenou spritu (obrázka) alebo animáciou. V hlavnom menu sme použili sfarbenie tlačidla. Highlighted Color je farba tlačidla v momente keď sa kurzor nachádza na tlačidle alebo je inak zvýraznené. Túto farbu sme zvolili svetlo sivú. Pressed Color je farba po stlačení tlačidla. Túto farbu sme nastavili na tmavšiu sivú aby bolo vidno rozdiel.

V scéne voľby levelu je použitý prechod ,,Sprite Swap“ čo je zmena obrázku. Vo finále to vyzerá akoby po zvýraznení obrázok stmavol a dostal orámovanie. V skutočnosti sa vymení jeden obrázok s ďaľším nami upraveným.

C:\Users\adamk\AppData\Local\Microsoft\Windows\INetCache\Content.Word\dok1.jpg

**12** Tlačidlo z hlavného menu pre výber levelu

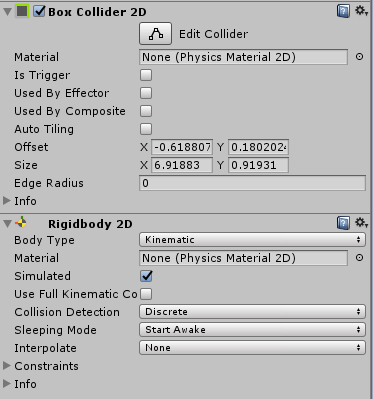


**13** Prechod po prejdení myšou vo výbere levelu

Nastavenie je podobné ako pri sfarbení. Namiesto farby sa vkladá nový sprite (obrázok). Najprv sme chceli použiť len ztmavenie obrázka no pre lepšiu viditeľnosť sme zvolili aj oranžové orámovanie, ktoré korešponduje s farbou písma v celej hre a drží sa teda jedného dizajnu.

# Tvorba levelov

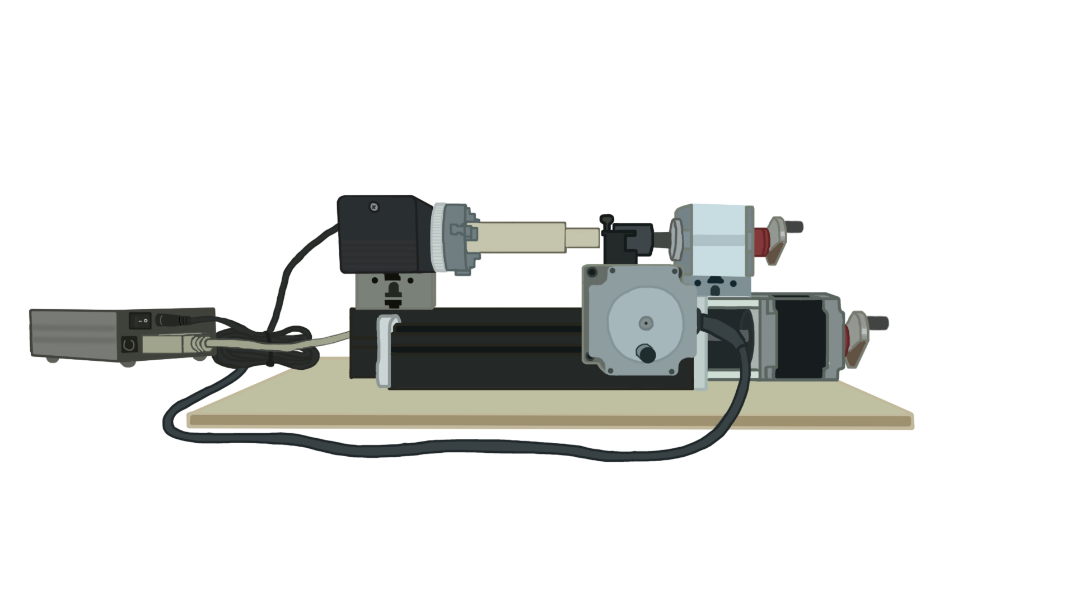
Keď už sme mali všetky modely nakreslené začali sme s tvorbou levelov. Modely vo formáte JPEG alebo PNG sme importovali do prostredia Unity. Tu sme im priradili komponenty ako ,,Box Collider 2D“, ktorý slúži na to aby postavy nemohli prejsť cez predmety keď sa dostanú do kontaktu. Ďalší dôležitý komponent je ,,Rigidbody 2D“. Tento komponent zapríčiňuje, že model sa po kontakte s postavou nepohne a zostane teda statický. Takisto naň nevplýva gravitácia.



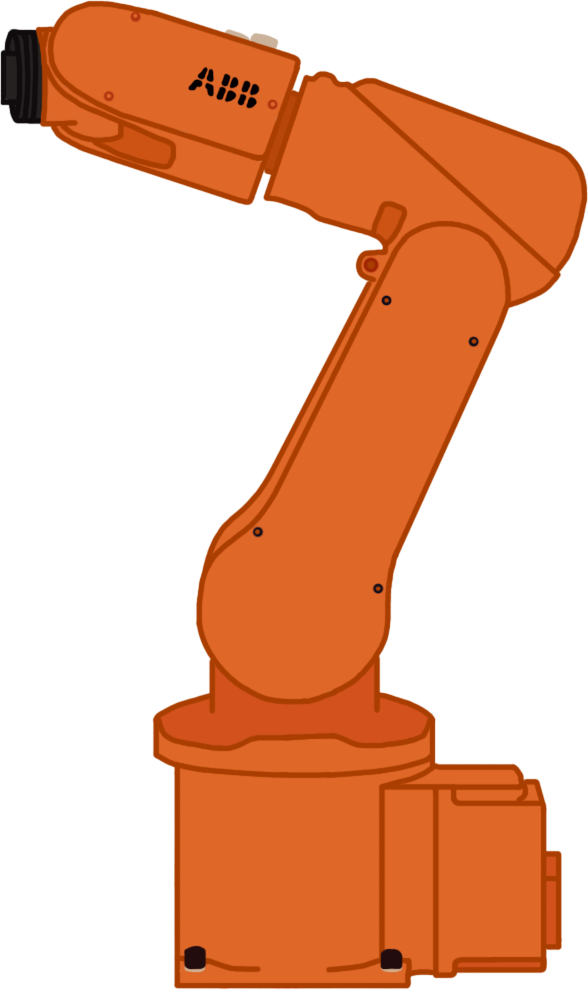
**14** Komponenty Box Collider 2D a Rigidbody 2D

Pomocou nastavenia ,,Edit Collider“ v komponente Box Collider 2D sa mení ohraničenie časti modelu, ktorá bude pre postavy neprechodná. Na takýchto platformách sa nachádzajú peniaze. Platformy musia byť pevné a pre hráča neprechodné aby po nich mohla postava behať a skákať. Levely v multiplayeri sú tématicky zamerané každý na iný odbor.

Prvý level sa zaoberá odborom elektrotechnika. V tomto leveli sú použité pomôcky, ktoré sa používajú najmä na výuku tohto odoru. Hlavnú časť levelu tvorí ,,VarioLab“. Ako plošiny sú tu použité elektrotechnické súčiastky ako rezistory a kondenzátory.

Pre strojársky level sú použité pomôcky ako napríklad model sústruhu, ktorý máme v škole a rôzne súčiastky súvisiace so strojárstvom. 

**15** Model sústruhu

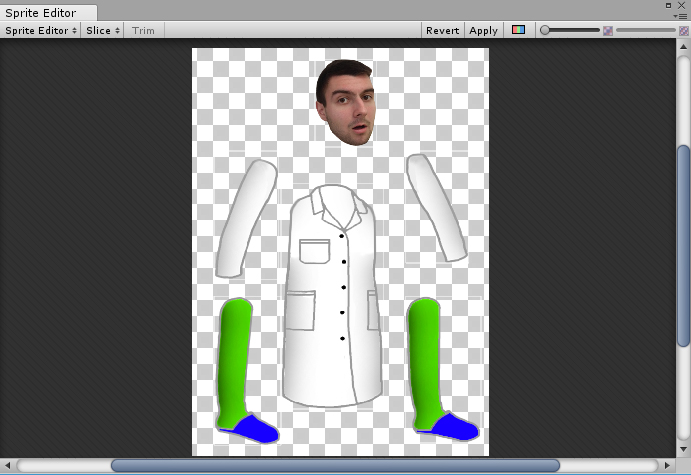
 V leveli na stole môžeme nájsť ozubené kolesá a matice, na ktorých sa zväčša nachádzajú peniaze. Niektoré matice sú umiestňené tak aby sa nedalo prejsť pred (pod) nimi ale hráč ich musel preskočiť. Levitujúce hriadele nesú najviac peňazí aby motivovali hráča skákať aj vyššie a aby bol odmenený hráč, ktorý je šikovnejší v skákaní.

**16** Herný objekt - robot ABB IRB120

Tretí level odboru mechatronika má niečo z oboch predchádzajúcich levelov. Nájdeme tu aj sústruh, ktorého programovanie majú mechatronici v osnovách. Sú tu aj elektrotechnické súčiastky nakoľko mechatronika je veľmi úzko spätá práve s elektrotechnikou. Navyše je tu však napríklad robot IRB120, ktorý by mal byť hlavne k dispozícii mechatronikom.

Postavy stvárňujú dvoch študentov, konkrétne nás, tvorcov tejto hry. Oblečení sú v bielom plášti, ktoré sme používali na prax. Nakoľko sú obaja v rovnakom plášti tak ich rozlišujú nohavice. Farba nohavíc oboch postáv je odlišná. Postavy majú reálne hlavy nás tvorcov aj naše mená. Chceli sme sa takýmto spôsobom dostať do hry a ozvláštniť t ak atmosféru v hre.

Postavy sme animovali cez Unity. Najprv bolo potrebné nakreslenú postavu rozdeliť na časti do jedného obrázku. Ten sme potom importovali do Unity a pomocou *Sprite Editora* sme časti pomenovali.



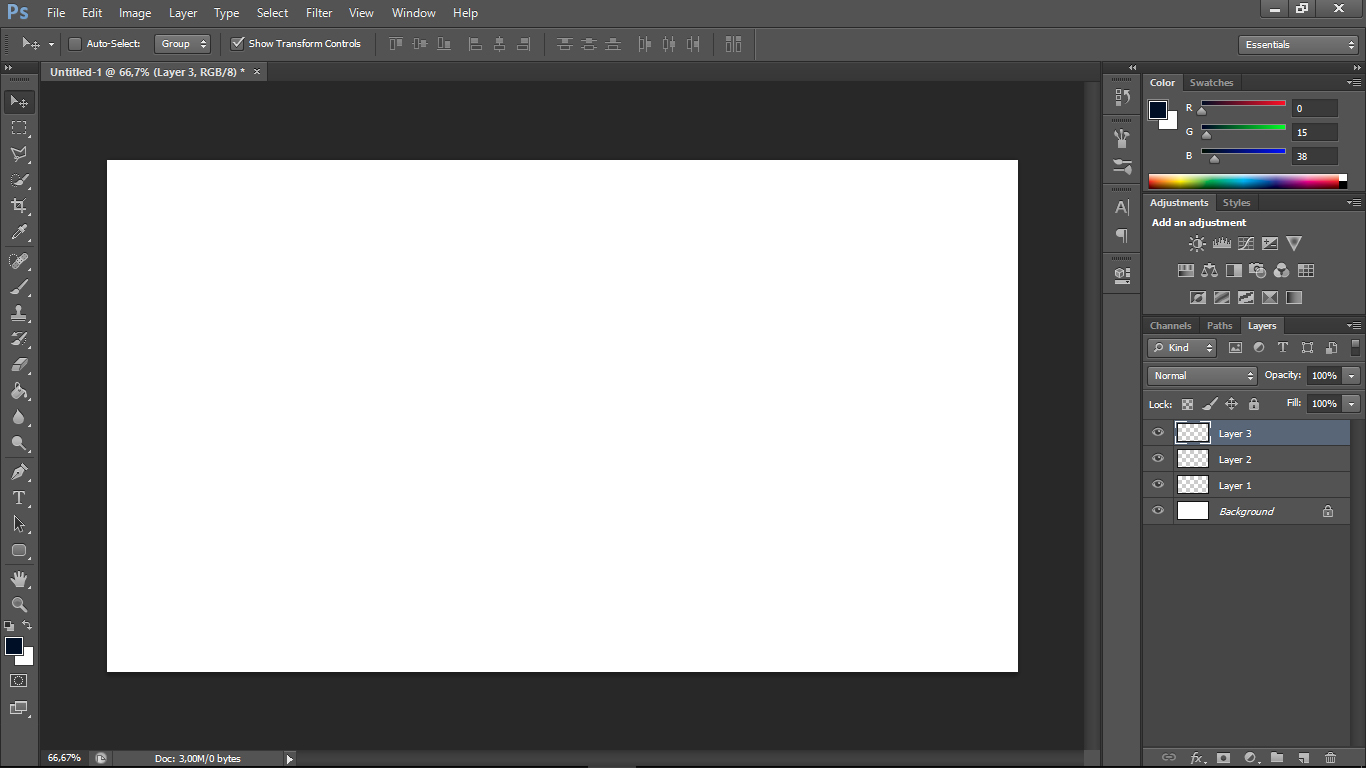
**17** Sprite editor - postava určená na animáciu

Takto vytvorené a pomenované časti sme poskladali. Objektu postavy sme pridali komponent *Animator.* V okne *Animation* sme pridali animáciu. Kliknutím na časovú os a pohybom častí sme nastavili každý snímok animácie. Týmto spôsobom sme vytvorili animáciu na pohyb, státie postavy a iné.

# Vytváranie grafických prvkov v programe Adobe Photoshop

Adobe Photoshop CS6 je grafický program od firmy Adobe Systems. Táto americká spoločnosť tvorí multimediálny a kreatívny softvér. Photoshop je hlavne určený na tvorbu a úpravu rastrovej grafiky. V programe sa tiež dá editovať vektorová grafika, ktorá sa dá exportovať do rastrovej. Ja som používal len rastrovú grafiku. Je veľmi používaný u profesionálnych grafikov. Celosvetovo má viac než 10 miliónov používateľov. Podporuje väčšinu obrazových súborov najmä ako JPEG, JPG, PNG (Portable Network Graphics-vie pracovať s transparentnosťou) ale aj mnoho ďaľších. Ja som najčastejšie používal formát PNG práve kvôli transparentnosti.

# Prostredie programu Adobe Photoshop



**18** Prostredie programu Adobe Photoshop

Na ľavej časti obrazovky sa nachádzajú nástroje, ktoré neskôr podrobnejšie opíšeme. Hore v záložkách ako File a Edit sú ďalšie podrobnejšie nastavenia a možnosti napríklad ako nastavenia súboru, krok vzad(CTRL+Z), nastavenie vrstiev a pod. Hneď pod nimi sa nachádzajú rozšírené nastavenia pre aktívny nástroj. Ešte nižšie môžeme nájsť kartu s názvom projektu a pod ňou pracovnú plochu konkrétneho projektu. Na pravej strane (zhora) sú nastavenia farieb, saturácie,... Vpravo dole sa nachádzajú veľmi dôležité vrstvy. Práve práca s vrstvami je vo Photoshope dôležitá. Vrstvy nie len že uľahčujú prácu no veľa vecí by nebolo možných bez nich.

Adobe Photoshop obsahuje množstvo nástrojov. Medzi nástrojmi sa samozrejme dá prepínať aj pomocou klávesových skratiek. Medzi najčastejšie nami používané nástroje patria:

**C:\Users\adamk\AppData\Local\Microsoft\Windows\INetCache\Content.Word\rectangular.jpg Rectangular Marquee – obdĺžniková selekcia**

Slúži na označovanie a selektovanie v pravidelno obdĺžnikovom tvare. Tento výber sa dá neskôr vyrezať, vyplniť farbou,...

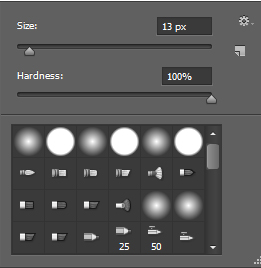
**C:\Users\adamk\AppData\Local\Microsoft\Windows\INetCache\Content.Word\move.jpg Move – presúvanie**

Move je nástroj slúžiaci na presúvanie selekcií a vrstiev.

**C:\Users\adamk\AppData\Local\Microsoft\Windows\INetCache\Content.Word\lasso.jpg Polygonal Lasso – mnohouholíková selekcia**

Po kliknutí ťahá úsečky až pokým sa nevráti kurzor do prvého bodu a vzniknutý mnohouhoľník vieme neskôr vypĺňať farbou ako pri obdĺžnikovej selekcii

**C:\Users\adamk\AppData\Local\Microsoft\Windows\INetCache\Content.Word\crop.jpg** **Crop Tool – na orezanie obrázku**

Orezáva celý obrázok a tým aj mení jeho rozmery. Crop Tool som používal najmä pri vytváraní menších plošín a obrázkov, ktoré som ukladal vo formáte PNG.

C:\Users\adamk\AppData\Local\Microsoft\Windows\INetCache\Content.Word\eraser.jpg **Eraser – gumovanie**

Guma klasicky maže. Dá sa jej meniť veľkosť, tvrdosť, tvar,..

C:\Users\adamk\AppData\Local\Microsoft\Windows\INetCache\Content.Word\brush.jpg **Brush – štetec na maľovanie**

Štetec som používal asi najčastejšie. Takisto ako pri gume sa dá meniť jeho veľkosť, tvrdosť a tvar. Pri štetci sa dá navyše meniť aj farba.

**C:\Users\adamk\AppData\Local\Microsoft\Windows\INetCache\Content.Word\bucket.jpg** **Paint Bucket – výplň selekcie alebo danej oblasti farbou**

Tento nástroj vypĺňa farbou selekcie. Veľmi urýchľuje prácu tým, že vypĺňa uzavretý priestor.

**C:\Users\adamk\AppData\Local\Microsoft\Windows\INetCache\Content.Word\type.jpg Horizontal Type – písanie vodorovného textu**

Vytvára textové pole, ktoré sa dá neskôr editovať.

Všetky grafické modely sme kreslili pomocou myši a Brush Tool. Zvažovali sme kúpu grafického tabletu no pre nedostatok financií sme zvolili túto metódu, ktorá je trochu zdĺhavejšia. Vzhľadom na naše skúsenosti a vybavenie sme sa nepúšťali do zložitejšej grafiky, nakoľko všetko v hre je vytvorené nami autormi. Ako predlohu a inšpiráciu sme používal prístroje, ktoré máme v škole a reálne na nich aj pracujeme. Plošiny, postavy a všetko v hre by malo súvisieť so zameraním školy a jej odbormi. Už na prvý pohľad by malo byť jasné o model akého prístroja alebo súčiastky sa jedná. Hra by mala vystihovať odbory Elektrotechnika, Strojárstvo, Mechatronika. Robot IRB120, ktorý je použitý v leveli Mechatronika, je naanimovaný a robí jednoduché úkony. Takisto prvky v menu sú kreslené vo Photoshope. Menu má mať jednoduchý vzhľad aby bolo čo najjednoduchšie a najintuitívnejšie. Nechceli sme žiadne graficky komplikované menu, ktoré pôsobí mätúco a lacno.

# Výsledky a diskusia

Unity sa ukázalo ako veľmi dobrý nástroj na tvorbu hier a je to veľmi dobrý nástroj pre vstup do sveta hier, respektíve do sveta vytvárania hier. Poskytol nám intuitívne prostredie pre tvorbu, odlaďovanie a spájanie herných komponentov. Poskytol nám základy a vysvetlenia pre knižnice Unity API.

Vďaka tejto práci sme sa toho mnoho naučili o vytváraní hier. Zistili sme ako je veľmi časovo náročné vytvoriť hru a koľko úsilia sa musí vynaložiť pre dosiahnutie výsledku ktorý uspokojí naše potreby a tak isto ukáže naše odborné znalosti. Táto skúsenosť nás ešte viac utvrdila v tom čo chceme robiť a čo vieme robiť a čo nás zaujíma. Táto práca nás naučila pracovať v dvojici, respektíve v tíme, komunikovať medzi sebou a pracovať efektívne pre dosiahnutie spoločného cieľu.

Ďalej by sme povedali že nás tento projekt obohatil o cenné skúsenosti ktoré nám už nikto nezaprie, naučili sme sa riešiť problémy efektívne, a že ich bolo mnoho. Naučili sme sa lepšie ovládať softvér Unity ako celok ale aj lepšie pracovať v použitých programoch a v neposlednom rade nám zlepšilo naše programovacie schopnosti. Tieto skúsenosti sú cennou odmenou do nášho budúceho života a pre našu kariéru.

Našu prácu sme si veľmi skomplikovali hlavne rozhodnutím že táto hra bude pre dvoch hráčov. Je veľmi náročné vytvoriť takúto hru ak ju porovnáme s hrou pre jedného hráča ako aj hrou v 3D zobrazení. Najťažšie na tejto hre bolo vytvoriť systém vyhodnocovania skóre pre dvoch hráčov a tým pádom aj vyhodnocovanie hráča ktorý vyhral. Pre túto funkciu sme museli použiť až 3 kódy, kdežto pri hre pre jedného hráča by stačilo použiť jeden zdrojový kód.

Samozrejme, 3D zobrazenie je lákavejšie na pohľad ale kreslenie všetkých herných objektov ručne, pomocou myšky bez grafického tabletu, je oveľa časovo náročnejšie a chce to aj určitú dávku trpezlivosti a predstavivosti. 3D modelovanie je ľahšie v mnohých aspektoch. Ako príklad by sme uviedli samotné modelovanie. V mnohých programoch pre tvorbu 3D objektov len stačí zadať súradnice alebo si nakresliť náčrt a program samotný objekt vymodeluje. Tak isto aj použitie farieb pri poloprofesionálnej práci môže byť fádne a celkovo nudné. Často kráť sa môžeme stretnúť že takéto hry majú všetky objekty hranaté, bez zaoblení čo v konečnom výsledku pôsobí tiež fádne.

Naša práca je obrazom nás a našich vedomostí. Keďže je určená pre dvoch hráčov, dokáže ľudí spájať aj keď im dodáva trochu rivality pri hraní. Po grafickej stránke naša hra nepôsobí fádne ale práve naopak. Hra je farebná a živá navyše obsahuje 2D animácie.

# ZÁVERY PRÁCE

Našim cieľom bolo vytvoriť 2D arkádovú hru pre dvoch hráčov so systémom vyhodnocovania skóre a tak isto aj možnosťou hry pre jedného hráča v endless móde.

Timotej Kusy bol zodpovedný za kódovú časť vytvárania tejto hry ako aj za testovanie týchto kódov a ich efektívnosť. Istou mierou zasahoval aj do Unity keďže veľa častí kódov sa prideľovalo v Unity a tak isto aj ich funkcie a zadávali sa hodnoty premenných mimo kódu. Svoje postrehy a problémy konzultoval s Mgr. Dušanom Horváthom ktorý ho navádzal správnym smerom.

Adam Kysler bol zodpovedný za audiovizuálnu stránku hry bez ktorej by hra bola len kódom v dokumente. Pracoval v prostrediach troch softvérových nástrojov, konkrétne v Adobe Photoshop, kde vytváral všetky grafické prvky, v programe FL Studio, kde vytváral hudbu pre našu hru a tak isto pracoval aj v Unity kde vytváral animácie a spájal všetky tieto komponenty aby dosiahol náš cieľ. Samozrejme že svoje nápady konzultoval s Timotejom Kusym a spoločne vždy dospeli k výsledku ako a čo v hre spraviť aby bola hra pútavá. Mgr. Dušan Horváth s ním taktiež všetko konzultoval a vďaka pravidelným stretnutiam vyjadril svoj názor čo a kam pridať aby hra bola pútavejšia.

Zadanie sa nám podarilo splniť. Hra funguje tak ako sme to chceli a obsahuje všetky zadané prvky, okrem nahrávania hry. Nahrávanie hry sme sa rozhodli nepridať pretože pri hre takéhoto formátu nemá žiadny zmysel, opodstatnenie a využitie. Skóre je efektívne vyhodnocované, pohyb postáv je plynulý a pri hraní sme nespozorovali žiadne problémy.

Animácie v hre fungujú a sú pútavé. Všetko čo sa má hýbať sa hýbe a tak isto neboli spozorované žiadne problémy.

Podarilo sa nám vytvoriť hudbu dotvárajúcu celú atmosféru hry. Nepôsobí nudne a fádne ale vytvára atmosféru typickú pre hry takéhoto typu.

V budúcnosti plánujeme hru rozvinúť o viac levelov a viac herných mechaník. Plánujeme pridať viac pútavých animácií, herných a grafických objektov ako aj preklad do ďalšieho jazyka a pridanie viac hudobných vzorov. V pláne máme tak isto pridať aj 3D herné objekty ktoré by boli naanimované a z pozadia by vystupovali čiže hra by sa stala 2.5D hrou. Hru tak isto plánujeme oživiť a ozvláštniť pomocou svetla. Rôzne farby svetla, rôzne intervaly blikania ako aj vysvietenie plošín po tom čo na ne hráč/ hráči stupia.

Build pre Android alebo Web ako aj odladenie pre tieto platformy je možný a zvažujeme ho ale až keď pridáme hore uvedené veci. Unity je v tomto veľmi dobré pretože dokáže buildovať na rôzne platformy ale je na nás, ako hru odladíme. Tak isto budeme musieť upraviť ovládanie pre Android nakoľko platforma Android je primárne určený pre mobilné zariadenia akými sú tablety a mobilné telefóny.

# ZHRNUTIE

Cieľom práce bolo vytvoriť 2D arkádovú hru pre dvoch hráčov s prepracovaným systémom vyhodnocovania skóre. Hra obsahuje 4 levely z toho jeden je pre jedného hráča a sú v ňom neustále a nekonečne generované plošiny.

V metodike sú opísanie herné mechaniky vytvorené pomocou kódov ako je napríklad pohyb alebo zber a vyhodnocovanie skóre. Ďalej sa v metodike opisuje vytváranie audiovizuálnej stránky. Dokumentácia obsahuje aj opis použitého softvéru a dôvody pre vytvorenie tejto práce ako aj dôvody nášho výberu žánru a použitého softvéru hry.

Zadanie sa nám podarilo splniť, hru sme dokončili a funguje presne tak, ako sme chceli aby fungovala.

# RESUME

The aim of this work is to develop an interactive game made in 2D graphics for two players. This game is of an arcade genre with 4 levels, one of the levels is endless singleplayer level. Game is developed in game engine Unity. Graphic objects are made in Adobe Photoshop and music is componed in FL Studio. Scripts are written in Microsoft Visual Studio using C# programming language.

This work gave us lots of experiences in game development and we better learned to use professional softwares. We learned that game development is mostly hard work and knowledge, also it needs to have an idea.

Game is working and we completed all given tasks.

# ZOZNAM BIBLIOGRAFICKÝCH ODKAZOV

Adobe Photoshop Help

<https://helpx.adobe.com/photoshop/user-guide.html>

Ben Tristem, Learn to Code by Making Games - Complete C# Unity Developer (kurz)

<https://www.udemy.com/unitycourse/>

Microsoft Visual Studio Support

<https://www.visualstudio.com/vs/support/>

Microsoft Word Support

<https://support.office.com/sk-sk/word>

Unity API

<https://docs.unity3d.com/ScriptReference/>

Unity Manual

<https://docs.unity3d.com/Manual/index.html>

Wikipédia – Adobe Photoshop

<https://en.wikipedia.org/wiki/Adobe_Photoshop>

Wikipédia – FL Studio

<https://en.wikipedia.org/wiki/FL_Studio>

Wikipédia – Microsoft Visual Studio

<https://en.wikipedia.org/wiki/Microsoft_Visual_Studio>

Wikipédia – Počítačová hra

<https://sk.wikipedia.org/wiki/Počítačová_hra>

Wikipédia – Unity (Game Engine)

<https://en.wikipedia.org/wiki/Unity_(game_engine)>

# ZOZNAM PRÍLOH

A – Zdrojové kódy (z dôvodu ich množstva a dĺžky sa nachádzajú ako samostatná príloha na CD)