Izrada 2D računalne igre

Žugaj, Antonio

SVEUČILIŠTE U ZAGREBU

FAKULTET ORGANIZACIJE I INFORMATIKE

V A R A Ž D I N

Antonio Žugaj

IZRADA 2D RAČUNALNE IGRE

ZAVRŠNI RAD

Varaždin, 2022.

SVEUČILIŠTE U ZAGREBU

FAKULTET ORGANIZACIJE I INFORMATIKE

V A R A Ž D I N

Antonio Žugaj

JMBAG: 0016136491

Studij: Informacijski sustavi

IZRADA 2D RAČUNALNE IGRE

ZAVRŠNI RAD

Mentor/Mentorica:

Izv. prof. dr. sc. Mario Konecki

Varaždin, srpanj 2022.

Antonio Žugaj

Izjava o izvornosti

Izjavljujem da je moj završni/diplomski rad izvorni rezultat mojeg rada te da se u izradi istoga nisam koristio drugim izvorima osim onima koji su u njemu navedeni. Za izradu rada su korištene etički prikladne i prihvatljive metode i tehnike rada.

Autor/Autorica potvrdio/potvrdila prihvaćanjem odredbi u sustavu FOI-radovi

Sažetak

U ovome radu će biti pojašnjen koncept dvodimenzionalne arkadne igre. Igri će biti opisan koncept i njene funkcionalnosti. Prije toga će biti opisani svi alati koje smo koristili. Oni su Apache server, GameMaker Studio i MySQL. Pojasnit ćemo zašto moramo koristiti te alate i na koji način ih zapravo koristimo. Prikazat ćemo nekoliko primjera dvodimenzionalnih i arkadnih igara i objasniti na koji način su one i dalje konkurentne modernijim igrama. Na kraju ćemo prikazati implementaciju najbitnijih funkcionalnosti igre. Funkcionalnosti će biti opisane riječima i pomoću dijelova koda.

Ključne riječi: računalna igra, 2D igra, GameMaker, game engine, arkadna igra, programiranje

Sadržaj

1. Uvod

Iako je industrija izrade računalnih igara jedna od najvećih industrija na svijetu, većina ljudi i dalje nije svjesna koliko je zapravo velika i koliko brzo raste. Svi smo prilikom odrastanja bili izloženi barem jednom tipu računalnih igri. Od starih arkadnih igara do suvremenih ekstremno realističnih. One su postale dio normalnog života kao izvor relaksacije i zabave. Teško je pronaći kućanstvo koje nikad nije posjedovalo barem jednu video konzolu ili osobno računalo na kojem su se nalazile računalne igre. Nije ni čudo kako su napretkom osobnih računala i brzine interneta računalne igrice dobile popularnost diljem svijeta i generiraju 150 milijardi USD godišnjeg prihoda.[1]

Kroz dugu povijest računalnih igara nastali su različiti žanrovi. Među prvim koji su nastali su arkadne igre i simulacijske igre. Nakon njih je nastao široki spektar različitih žanrova kao što su: sportske igre, igre avanture, igre pucača u prvom licu i igre uloga. Zbog limitirane tehnologije u ranim početcima računalnih igri, sve su igre bili dvodimenzionalne, ali laganim napretkom tehnologije svake godine su postajale sve realnije. Današnje igre su velikom većinom trodimenzionalne, ali to ne znači da ne postoje dvodimenzionalne i da nisu popularne. Neki od primjera takvih igara su „Flappy bird“, „Undertale“ i trenutačno popularan „Among us“. Iako te igre u grafičkom smislu nisu ni blizu konkurenciji, one su skupile veliku popularnost i podršku velikog broja ljudi.

U ovom radu ćemo opisati postupke izrade jedne dvodimenzionalne igre arkadnog žanra. Prikazat ćemo korištenje GameMaker Studio 2 engine-a, izradu grafičkih i audio resursa, te način spajanja na server, odnosno poslužiteljsko računalo. Opisat ćemo koncept igre i razjasniti implementaciju ključnih funkcionalnosti.

2. Alati i programski jezici

Kako bi napravili igru moramo odabrati takozvani engine, odnosno program za kreiranje igre. Engine kod igra kreira lako definirano razdvajanje između njegove jezgre softverskih komponenti (npr. Iscrtavanje trodimenzionalnih grafika, detekcija sudaranja i audio sustava) i grafičkih resursa, svjetova i pravila igre koja daju korisničko iskustvo.[2] Iako ima puno popularnih engine-a, ja sam odabrao koristiti GameMaker Studio 2. Taj engine je kreiran za izradu dvodimenzionalnih igara i lagan je za korištenje. Drugi razlog korištenja baš tog engine-a je moje prijašnje iskustvo s njime. Pošto unutar igrice imamo globalnu listu najboljih rezultata, moramo imati centralizirani zapis svih rezultata. Za to nam je potreban server, odnosno poslužiteljsko računalo. Za ovaj rad smo odabrali Apache server. Zapis na tom serveru, radi lake organizacije, zapisujemo u MySQL bazu podataka. Kako bi kreirali igru, moramo znati alate i sučelja koja su nam potrebna. U sljedećim paragrafima ćemo pojasniti sve nabrojane alate i tehnologije.

2.1. Alati

2.1.1. Apache

Apache je open source HTTP server za UNIX i Windows. Open source je i ima veliku potporu. Cilj Apache-a je osigurati siguran, učinkovit i proširiv poslužitelj HTTP usluge u skladu s trenutnim HTTP standardima.[3] HTTP (Hyper Text Transfer Protocol) je protokol za prijenosa podataka i informacija na Web-u. Uz kombinaciju s mogućnostima DBMS-a, koje Apache nudi, postao je najpopularniji server na internetu. DBMS (Database managment System) je sustav upravljanja bazom podataka. Najpopularniji je MySQL koji se također koristi u ovome radu.

2.1.2. MySQL

MySQL je jedan od najpopularnijih sustava za upravljanje bazom podataka. Open source, što znači da ima veliku potporu i konstantna nadograđivanja. Relacijskog je tipa i ima potporu za razne programske jezike. Među kojima je i PHP.

Zašto koristiti bazu podatka umjesto nekog drugog načina zapisa podataka? Korištenje baze podataka znači brzo pohranjivanje i pronalaženje informacija. Korisnici i aplikacije imaju brzi način asinkronog čitanja i pisanja podataka.[4] To su glavni razlozi odabira korištenja baze podataka poput MySQL-a. Pri tome uz SQL jezik možemo postaviti određen uvijete nad podacima koje dohvaćamo. U našem slučaju trebamo dohvatiti ljestvicu 10 najboljih vremena, zajedno s korisničkim imenom. MySQL nam daje mogućnost da s jednim jednostavnim SQL upitom dohvatimo samo one podatke koji su nam potrebni i sortirani prema zahtjevima.

2.1.3. GameMaker Studio 2

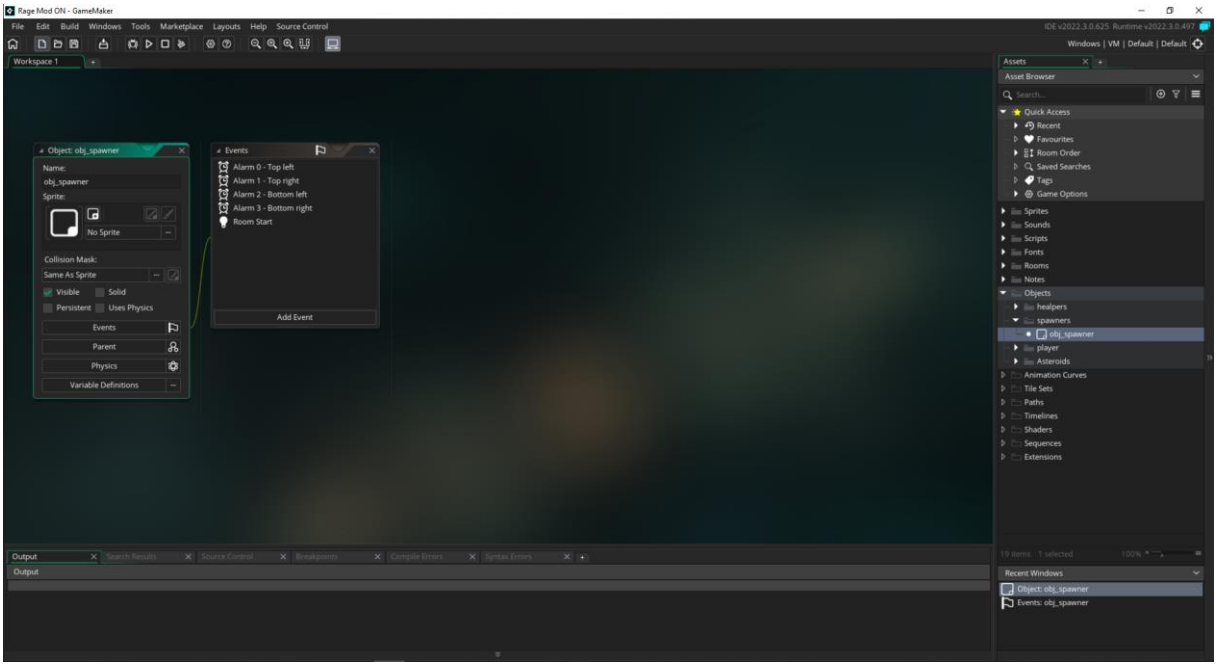
GameMaker Studio je linija više-platformskih engine-a koji se baziraju na dvodimenzionalnim igrama. GameMaker Studio 2 je zadnje izdanje programa koje je izašlo 2017. godine. Objavila ga je kompanija YoYo Games, koja ga objavljuje i nadograđuje od 2007. godine. Originalno ga je kreirao Mark Overmars. Program je prvi put objavljen pod imenom Animo, 15 listopada 1999.[5] Nakon toga je ime promijenjeno u GameMaker. Engine se nadograđivao i postajao sve popularniji i popularniji. Mnoge poznate igre su kreirane koristeći GameMaker. Neke od njih su „Hotline Miami“, „Spelunky“, „Hyper Light Drifter“, a među njima je najpoznatija „Undertale“ koja je primila ogroman broj nagrada i uspjeha. Do danas GameMaker proizvodi su bili preuzeti više od deset milijuna puta od 2012. godine i trenutno imaju 200 000 aktivnih mjesečnih korisnika.[6] GameMaker daje mogućnost kreiranja igara na više platformi. Najkorištenija je naravno Windows, ali imaju mogućnosti i kreiranja za macOS, Ubuntu, HTML5, Android, IOS, Playstation 4, Xbox One i Nitendo Switch.

Za razliku od većine engine-a rane verzije GameMaker-a su bile fokusirane na početnike u programiranju i izradi računalnih igara. GameMaker ne koristi već kreiran i poznat programski jezik. GameMaker koristi svoj interni jezik pod nazivom GameMaker Language. Postoje dvije varijante tog jezika. Prva je grafička strana, koja daje mogućnost „programiranja“ bez pisanja koda. Druga je kodna strana, koja daje veću kontrolu i vlastito pisanje koda. Originalno je veći fokus bio na grafičkoj strani. GameMaker Studio se reklamirao kao odlična

prva platforma za kreiranje računalnih igara. Pomoću grafičkog jezika korisnik nije trebao poznavati programiranje i zbog dvodimenzionalne prirode igara nije bilo teško pohvatati konce.

U sljedećim verzijama GameMaker-a veći fokus je bio dan na iskusnim programerima. Povećale su se mogućnosti kontroliranja događaja pomoću koda. U ranijim verzijama se do neke mjere morao koristiti grafički GameMaker jezik, dok u najnovijoj verziji GameMaker Studio 2, on se u potpunosti može isključiti. Pisani GameMaker jezik je veoma jednostavan za korištenje i za razumijevanje.

GameMaker Studio 2 je jako jednostavan za korištenje. Kod otvaranja programa se otvara površina za rad u obliku kartice. Kod uređivanja više različitih stvari, radi organizacije, se predlaže otvaranje dodatnih kartice. Dodatne kartice se otvaraju pri otvaranju sobe ili scene. S desne strane programa nalazi se odjeljak u kojem su izlistani svi resursi kojima upravljamo. Postoje direktoriji za audio resurse (eng. Sounds), grafičke resurse (eng. Sprites), skripte (eng. Scripts), fontove, sobe (eng. Rooms), bilješke (eng. Notes), objekte (eng. Objects), animacije, puteve (eng. Paths), sekvence (eng. Sequences), sjenila (eng. Shaders) i ekstenzije (eng. Extensions). Pri otvaranju svakog od resursa nabrojanih u prijašnjim direktorijima, oni se prikazuju unutar otvorene kartice, osim ako se ne radi o sobama. Donja strana sučelja je namijenjena za prikaz pogreški ili problema kod kompiliranja i izgradnje igre. Na samom vrhu sučelja se nalaze alati za kompiliranje, testiranje, spremanje i postavljanje postavki igre. Izgled sučelja programa može se vidjeti na slici: Sučelje GameMaker-a.



Slika 1. Sučelje GameMaker-a

Najvažniji element kod izrade računalne igre su objekti. Njihova interakcija kreira dinamiku igranja i stvara razliku između animacije i igre. Ako bi računalnu igru na najprimitivniji način opisali, onda bi ona bila niz slika koje se svojim dodirivanjem ili nekakvim unosom korisnika mijenjaju. Te slike su zapravo objekti. Objekti sadrže ime, grafički resurs, koji ga predstavlja, masku za kolizije, događaje, roditelje i fiziku.

Objekt na sebe veže grafički resurs koji je unaprijed pripremljen. Objekt preuzima veličinu i oblik grafičkog resursa. Maska kolizije je zamišljena površina koja aktivira određene događaje ako se maska kolizija dva različita objekta dodiruje. Primarno je istog oblika i veličine kao i grafički resurs, ali se može posebno namjestiti. Svaki objekt može biti roditelj drugom objektu i svaki objekt može biti dijete nekom drugom objektu. Kod stvaranje veze roditelj-dijete sve podatke i skripte vezane za događaje se prenose sa roditelja na dijete. Unutar djeteta se automatski te skripte aktiviraju, ali u slučaju da dijete ne treba naslijediti neki događaj ili ga djelomično naslijediti, onda se može izbrisati, nadodati ili potpuno izmijeniti. GameMaker Studio nudi svoja vlastita pravila za fiziku. Ona su namijenjena primarno za početnike. Za iskusnije programere se uvijek predlaže da napišu svoja pravila za fiziku i interakciju između objekata, kako bi imali veće razumijevanje i kontrolu.

Najbitniji element objekata su događaji. Oni od statičnih nepromjenjivih objekata kreiraju interaktivne dinamičke objekte. Svaki događaj ima skriptu koja se izvršava pri svakom pozivanju tog događaja. Postoje razne vrste događaju. Neke od njih su stvaranje (eng. Create), uništenje (eng. Destroy), korak (eng. Step), alarm, crtanje (eng. Draw), interakcija s mišem, pritisak tipkovnice, otpuštanje i držanje tipke tipkovnice, gesta (eng. Gesture), kolizija s drugim objektom, asinkroni signali, početak igre i sobe te izlazak iz sobe. Najbitniji događaji su događaj stvaranja i događaj korak.

Događaj stvaranja se izvršava samo jedanput kod svake instance objekta. On sadrži skriptu koja se izvrši kod kreiranja svake instance. Tipično se koristi za postavljanje početnih vrijednosti instance ili za generiranje varijabli potrebnih za ponašanje instance objekta. Na primjer u mojem radu se koristi za postavljanje početnih vrijednosti glavnog lika i kako bi generirali tip, rotaciju i smjer objekata koji se kreču oko glavnog lika.

Događaj koraka je događaj koji se neprestano izvodi. Frekvencija pozivanja je namještena u osnovnim postavkama igre. Poziva se svakim „okvirom“ (eng. Frame), što znači da se izvodi onoliko puta koliko ima „okvira“ po sekundi. U ovom događaju se izvršava promjena pozicija objekta, sluša se za unos korisnika, mijenjaju se animacije i provjerava se pravila fizike. Ovaj događaj je srž svakog objekta. Rijetki su objekti koji nemaju ovaj događaj.

Događaj Alarm se unutar nekog drugog događaja mora postaviti. Postavi se vrijeme koje se sa svakim „okvirom“ odbrojava. Kada alarm dođe do 0 on aktivira skriptu koja je zapisana unutar njega. Dok alarm nije postavljen ima vrijednost -1.

Kako bi objekti imali svoju instancu, moraju biti postavljeni u sobe. Sobe su grafičko područje unutar kojeg se može crtat i može vidjeti interakcija objekata. Ako bi uspoređivali s umjetnosti, sobe su platno, a objekti su boje. Unutar jedne igre može biti veliki broj soba, ali uvijek mora postojati barem jedna soba. Tipična praksa od video igara je da se glavni lik nalazi u jednoj sobi i nakon neke akcije ili događaja premjesti u drugu sobu. S time se dobije osjećaj napretka. U našem radu sobe koristimo kao prostor za različite izbornike i različite prostore za igranje ovisno o odabranom načinu igranja. Instance jedne sobe nestaju kod prelaska u drugu sobu, osim ako te instance nisu označene kao stalne (eng. Persistent).

Grafički se resursi mogu kreirati unutar GameMaker-ovog grafičkog uređivača ili se mogu uvesti iz vanjskih izvora. Kod kreacije se odabiru dimenzije, boje i oblik slike. Slici se postavlja centralna točka i maska. Centralna točka i maska se tipično koriste za slike, ali se i mogu unutar objekta promijeniti neovisno o slici.

Audio resursi se uvoze iz vanjskih izvora. Može se odabrati različiti načini reprodukcije, kvaliteta, frekvencija i kompresija. Može se, iako nije uvijek potrebno, grupirati audio resurse u određene grupe. U tom slučaju se mora pripaziti da li su te grupe pripremljene za puštanje. Grupiranje je poželjno radi mijenjanja volumena ili ponavljanja određenih audio resursa u odnosu na druge audio resurse.

Skripte su programski kodovi koji ne pripadaju niti jednom objektu. One su zaseban kod koji izvršava određenu zadaću. Pozivanje skripata mogu izvršiti sve instance svih objekata. Kreiraju se u slučajevima kada više objekata izvršava isti kod ili u slučajevima gdje želimo sakriti neki kod radi organizacije i čitljivosti. Unutar jedne datoteke skripti može se nalaziti više funkcija. Objekti ne moraju izvršiti sve funkcije. To je promjena od ranijih verzija GameMaker-a.

2.2. Programski jezici

Većina programskih jezika može se klasificirati u obitelji na temelju njihovog modela računanja. Deklarativni jezici usredotočuju se na upućivanje računalu što treba učiniti, dok se imperativni jezici usredotočuju na to kako bi računalo trebalo nešto napraviti. Programski jezici se također mogu kategorizirati na temelju toga imaju li sustav tipova ili ne. U jezicima s tipom, programske varijable imaju gornju granicu raspona vrijednosti koje mogu preuzeti. S druge strane, jezici bez tipa ne ograničavaju raspon varijabli.[7] To znači da u jezicima bez tipa varijable mogu mijenjati svoj tip bez ograničenja. U izradi ovog rada koristili smo čak dva jezika koji nemaju tipove. To su PHP i GML (GameMaker Language). Oba programska jezika su interpreterskog tipa. Pošto koristimo bazu podataka, najčešće moramo koristiti i SQL jezik. Njega koristimo i u ovome radu radi dohvaćanja, sortiranja i filtriranja iz baze koja sadrži podatke o računalnoj igri.

2.2.1.SQL

SQL (Structured Query Language) je de facto standardni jezik sučelja za relacijske baze podataka od 1977. SQL nije potpuni programski jezik u smislu Pascal-a i C-a, ali je podatkovni pod jezik koji pruža mogućnost definiranja, učitavanja, održavanja i iskorištavanja relacijske baze podataka.[8] MySQL sustav za upravljanje bazom podataka također koristi SQL. U našem radu ne koristimo zahtjevne i komplicirane naredbe SQL-a. Koristimo naredbe za unos podataka (INSERT INTO) i za dohvaćanje podataka (SELECT). Uz njih koristimo i ORDER BY koji grupira zapise prema odabranom atributu, DESC koji nam sortira ispis po tom atributu i LIMIT koji nam limitira broj ispisa. Na taj način dobijemo listu podataka koji se ne treba dodatno programski sortirati i filtrirati.

2.2.2. PHP

PHP (Hypertext Preprocesor) je programski jezik koji se koristi na tri primarna načina. Koristi se za skriptiranje na strani poslužitelja, skriptiranje komandne linije i za GUI klijentske aplikacije.[9] Interpreterski tip programskog jezika koji nema tipove varijabli. Originalno kreiran za kreaciju dinamičkih web stranica. U ovome radu ne koristimo PHP za tu ulogu. PHP-e skripte koje se nalaze na Apache serveru čekaju pozivanje iz računalne igre. Nakon pozivanja koristimo skriptu za primanje podataka iz igrice, u slučaju zapisa, ili signala koji znači dohvaćanje podataka iz baze podataka. Umjesto kreiranja dinamičnog HTML, koristimo PHP za komunikaciju između igre i MySQL baze podataka.

Postoje tri pristupa kojima PHP upravlja bazom podataka. Postoji MySQLi objektno orijentiran način, MySQLi proceduralni način i PDO (PHP Data Objects) način. Pošto radimo rad s MySQL sustavom za upravljanje bazom podataka, preporučava se korištenje jedne od MySQLi metoda. Razlika između MySQLi objektno orijentirane metode i MySQLi proceduralne metode je, kao što i samo ime kaže, objekto kreira vezu kao objekt i operacije koje se izvršavaju nad bazom se izvršavaju kao metode tog objekta. Proceduralni sprema vezu u obliku varijable. Operacije u proceduralnom se izvršavaju kao funkcije nad varijablom u kojoj je spremljena veza.

Radi jednostavnosti i male količine PHP skripti u radu, koristimo MySQLi proceduralnu metodu. Iako ne bi bilo pogrešno koristiti i MySQLi objektno orijentiranu metodu. Još jedan razlog korištenja proceduralne metode je prethodno znanje i korištenje u sklopu srednjoškolskog i fakultetskog obrazovanja.

2.2.3.GameMaker Language

GameMaker Studio, za razliku od većine engine-a, ne koristi ranije kreiran programski jezik. Drugi engine-i koriste programske jezike koji nisu specijalizirani za taj engine. Naprimjer Unity engine može kreirati skripte koristeći C# programski jezik. Unreal engine koristi C++. Za razliku od njih GameMaker Studio engine koristi svoj programski jezik koji je kreiran isključivo za GameMaker Studio. Taj programski jezik se naziva GameMaker Language.

GameMaker Language je interpreterski programski jezik koji nema definirani tip varijabli. Kao najbliži vizualni opis, usporedio bi ga s JavaScript-om. GameMaker Language može se koristiti za programiranje igara na dva različita načina: Postavljanje blokova za vizualno „pisanje“ koda ili ručnim unosom instrukcija.[10] To su dva zasebna načina s kojim se može izvršiti isti cilj.

Grafički GameMaker jezik je namijenjen početnicima koji ne znaju dobro koncepte programiranja i programskih jezika. On daje opciju kreiranja grafičkih pomagala koji se povuku i puste na mjestima koja su pogodna za njih. Umjesto korištenja grananja, postoji već predefinirani blokovi koji predstavljaju ta grananja. Na ta grananja se dodaju različiti grafički objekti koji predstavljaju iteracije, inicijalizacije varijabli, postavljanje i promjena vrijednosti varijabli, interakcija s drugim objektima, unos preko tipkovnice ili miša itd. Iako je lagano raditi s grafičkim jezikom, on ne sadrži pravo programiranje. Zbog toga nije moguće imati potpunu kontrolu nad malim promjenama. Grafički modeli su generalni i ne daju mogućnost sve i jednog mogućeg ishoda. Grafički jezik se preporučuje koristiti onoliko dugo dok se ne razumije kako GameMaker funkcionira i dok se ne stekne iskustvo u programskim jezicima. Zbog manjka kontrole i ne kreiranja koda u ovom radu ne koristimo grafički jezik.

Za razliku od grafičkog jezika, GameMaker Language kod daje potpunu kontrolu nad varijablama, grananjima i iteracijama. Umjesto grafičkih blokova, dozvoljava nam korištenje ogromnog broja unaprijed definiranih funkcija, metoda i objekata. GameMaker Langage kod se primarno koristi unutar događaja objekata, ali može se koristiti i u kreiranju skripata kojima imaju pristup svi objekti. Programski jezik ima vrlo laganu i ima čitljivu sintaksu koja olakšava početnicima pronalazak odgovarajuće funkcije i metode koja im je potrebna.

Kao i većina programskih jezika, GameMaker Language se sastoji od varijabli, izraza, instrukcija i uvjeta. Blokovi programskog koda se moraju postaviti unutar vitičastih zagrada kako bi se prepoznali kao jedna logička cjelina. Može se koristiti i instrukcija „begin“ koja označava početak cjeline i instrukcija „end“ koja označava kraj cjeline. Nakon instrukcije nije potrebno postaviti točku zarez, ali se preporučuje. U slučaju da se koristi više instrukcija unutar

jedne linije se obavezo koristiti točka zarez. Komentari se označuju na isti način kao u većini programskih jezika. Dvostruka kosa crta za jednolinijski komentar i kosa crta i zvjezdica za više linijski komentar.

Postoje različite vrste deklaracija varijabli. Svaki tip deklaracije daje varijabli opseg u kojem joj se može pristupiti. Varijable mogu biti lokalne varijable, varijable instance, globalne varijable i konstante. Lokalna varijabla je ona koju kreiramo samo za određeni događaj ili funkciju, a zatim odbacujemo kada događaj ili funkcija završi. Ako je kreirana u vlastitoj funkciji, lokalna varijabla je dostupna samo funkciji, a zatim se odbacuje kada funkcija završi.[11] Ovakva varijabla ne zauzima nepotrebnu memoriju. Kako bi kreirali lokalnu varijablu moramo prije imena varijable staviti ključnu riječ „var“. Varijabla instance kreira se unutar instance objekta i smatra se jedinstvenom za tu instancu. Mnoge instance istog objekta mogu imati istu varijablu, ali svaka varijabla može imati različitu vrijednost jer su jedinstvene za svaku instancu. [12] Ovakve varijable se definiraju na način da samo upišemo ime varijable. Globalne varijable funkcioniraju isto kao u većini programskih jezika. Kada se deklariraju, ne pripadaju ni jednoj instanci i sve instance im imaju pristup u bilo kojem dijelu koda. Definiraju se tako da se upiše „global.“ i nakon toga ime varijable, bez razmaka. Konstante su varijable kojima se vrijednost ne može mijenjat nakon inicijalizacije. Definiraju se na ovaj način: „ #macro <ime varijable> <vrijednost>“.

Varijable tijekom svog postojanja mogu poprimiti više različitih tipova. Tipovi koji postoje unutar GameMaker jezika su: real number, bool, string, array, structs, method varijables, int64, hexadecimal values, pointer, enum, undefined, NaN, infinity, variable. Tip varijable koji može biti varijabilan (variable) se koristi kod funkcija i argumenata kada se ne zna kakav tip podataka će funkcija primiti. Beskonačni (infinity) tip podataka je rezultat operacija koje vračaju približno beskonačnu vrijednost. Na primjer dijeljenje s nulom.

3. Koncept video igre

Definiranje granica onoga što se naziva računalna igra ili video igra je kompliciranije nego što se čini. Ako raščlanimo riječ na „video“ i „igra“ možemo definirati posebno svaku stavku kako bi dobili cjelokupno sliku. Unutar igre se može očekivati nekakav konflikt, pravila, mogućnosti i konačan rezultat. Konflikt može biti protiv protivnika ili protiv određene situacije. Pravila definiraju što igrač može i što ne može napraviti kako bi razriješio konflikt. Mogućnosti igraća su sve sposobnosti koje igrač ima. One mogu biti strategija, sreća i motorika. Konačan rezultat je dio koji ovisi o uspješnosti rješavanja konflikta. Rezultat može biti pobjeda, gubitak, najbolje vrijeme i najviše bodova ili završetak narativne priče.[13] Kada se definicija igre poveže s videom, tada dobivamo računalnu igru. Igru gdje se svi elementi igre predstavljaju na grafički način na ekranu računala.

Bilo je potrebno dugo vremena kako bi televizija i film dobili status umjetnosti. Također računalnim igrama je isto dugo trebalo kako bi poprimili taj status.[13] U današnjim vremenima, iako se neki ne slažu, računalne igre su umjetnost na najvišoj razini. Umjetnost koja se može proživjeti i u kojoj svaki igrač može sudjelovati i na svoj način nadopuniti cjelokupni umjetnički aspekt. Svaka osoba drugačije percipira računalne igre, baš kao kod umjetnosti. Pravi primjer takve igre je „Undertale“, o kojoj ćemo kasnije pričati.

3.1. 2D računalne igre

Iako je cilj današnjim igrama postati što realističnije, u početcima su računalne igre bile par karikatura na obojanoj pozadini ekrana. Igre su bile smještene unutar dvije dimenzije. Moglo se kretati gore, dolje, lijevo i desno. Razlog tomu je bila limitirana tehnologija. Današnja tehnologija je poprilično napredovala, ali i dalje se kreiraju dvodimenzionalne igre. Postoji više razloga tome. Postoji lakoća kreacije, ali i izazov kreiranja naracije i dobrog iskustva u usporedbi s današnjim igrama. Drugi ih rade kako bi oživjeli nostalgiju iz mladosti. Nekima odgovara dvodimenzionalni medij u umjetnički koncept igre ili u zamišljena pravila igre, koja u trodimenzionalnom svijetu ne bi imala smisla. Neke od klasičnih tipova dvodimenzionalnih igra su igre vertikalnog ili horizontalnog pomaka, klasične arkadne igre, animirane avanture, zagonetke i popločene igre.[14] U našem slučaju kreirali smo dvodimenzionalnu arkadnu računalnu igru.

3.2. Arkadne igre

Prema Merriam-Webster rječniku, arkadna igra je igra koja se obično pokreće na novčić i dizajnirana je za igru u zabavnoj arkadi.[15] Arkade su bile prostorije koje su sadržavale razna računala na kojima su bile određene računalne igre. Kako bi igrač mogao igrati igru, morao bi pokrenuti igru s određenom količinom novca. Te igre su bile jednostavne i nisu imale detaljnu narativu. Cilj takvih igara je bila naravno zarada. Pokušali su postići što veći broj igranja. Igre su imale jednostavan zadatak. Pobjeći od duhova, utrkivati se protiv drugih, skupiti što više bodova u što kraćem vremenu i pobijediti računalo. Nažalost, arkade su nestale i s njima su nestala i računala s arkadnim igrama, ali ako pogledamo današnje tržište, arkadne igre i dalje postoje.

Usred rasta popularnosti arkadnih igara, u Americi i Japanu, pojavio se novi, ne konvencionalni arkadni sistem. Njega je dizajnirala Japanska kompanija SNK i zvao se Neo Geo Multi Video System (MVS). Njegova prednost je bila što je bio namijenjen za kućanstva i na sebi je sadržavao više arkadnih igara, za razliku od arkadnih računala koja su imali samo jednu igru.[16]

Nakon toga je krenula revolucija kućnih konzola i smrt arkada. Unatoč tome arkadne igre su preživjele. S napretkom tehnologije su napredovali i žanrovi računalnih igara. Nastale su igre koje se baziraju na narativne priče, koje se baziraju na vizualne spektakle i igre koje potiču socijaliziranje. Unatoč napretku, igrači su tražili žanrove igara kao što su bili u arkadama. Te igre su imale liste s najboljim igračima i najboljim vremenima. To je poticalo igrače da nastave igrat. Konkurencija je primaran razlog igranja i nastojanje potpuno savladati zahtjevne sposobnosti koje su potrebne za stvoriti novi rekord. Sustavi nagrađivanja, koji jasno stvara konkurenciju i uspoređivanje, su instrumentalni za izazivanje osjećaja angažmana, uključenosti i usredotočenosti pri igranju igre.[17] Takav sustav izrazito nagrađuje bilo kakav napredak u sposobnostima igrača. Zbog tog razloga su arkadne igre preživjele. Umjesto grupe prijatelja koja igra igre u arkada i natječe se to je najbolji među njima, današnje arkadne igre rade isto to, ali preko interneta. Gdje se natječu ne samo grupa prijatelja, već cijeli svijet. Jedna od najboljih primjera takve igre je „Flappy Bird“. To je igra koja je pokorila svijet 2013. godine. O njoj ćemo više detalja reći u sljedećim poglavljima.

3.3. Opis koncepta igre

Igra koju sam napravio u sklopu ovoga rada je dvodimenzionalna arkadna igra. Koncept igre sam osmislio još u srednjoj školi, gdje sam pokušao napraviti prvu verziju ove igre koristeći ranije verzije GameMaker Studio-a. Koncept je vrlo jednostavan, kao i rane arkadne igre.

Cilj igre je preživjeti što duže vremena. Igra nema narativnu priču. To joj nije fokus. Fokusirana je na kompetitivnu ljudsku prirodu. Glavni lik, kojega kontrolira igrač, je astronaut. Astronaut je u obliku male karikature astronauta u plavom svemirskom odijelu. Astronaut se nalazi u svemiru, ali je dovoljno blizu nekom neimenovanom planetu koji ga blagom gravitacijom vuče prema dnu ekrana. Oko igrača se na rubovima vidljivog prostora stvaraju asteroidi koji se rotiraju i putuju u nasumičnim smjerovima. Ako igrača dotakne asteroid ili ako igrač otiđe izvan vidljivog okvira igre, igrač završava igru. Dok igrač igra igru u gornjem lijevom kutu ekrana mu se broji vrijeme. Vrijeme odbrojava dok god je glavni lik u igri. Nakon što igra završi, prikazuje se vrijeme koje je igrač postignuo te se također prikazuje i najbolje vrijeme igrača. Igra s vremenom postaje sve teža i teža jer se svake četiri sekunde stvaraju novi asteroidi. Kada asteroid izađe izvan vidljivog okvira igre, ima dvadeset posto šanse da nestane iz igre. U suprotnom se asteroid vrača u vidljivi okvir igre i putuje u suprotnom smjeru (okrenut smjer za 180 stupnjeva). Kako bi igrač izbjegao asteroide, igrač ima način letenja. Pritiskom na tipku igrač se kreče u smjeru u kojem je nagnut. Kada prestane držati tipku, gravitacije ponovno počinje djelovati. Kako bi olakšali izbjegavanje asteroida, igrač ima mogućnost promjene kuta. Tim naginjanjem igrač kontrolira u kojem smjeru leti i u kojem smjeru pada. Kako bi otežali igru i stvorili potrebu za većim vještinama, dodali smo limit na letenje. Držanjem tipke za letenje se puni linija koja predstavlja pregrijavanje. Ako pregrijavanje dosegne maksimum, igraču se onemogući mogućnost letenja. To onemogućenje traje kratak vremenski period, ali je dovoljno velik da zakomplicira igru igraču. Dok igrač igra u pozadini se čuje retro 8-bit-na glazba. Prikaz izgleda igre se može vidjeti na slici 2.

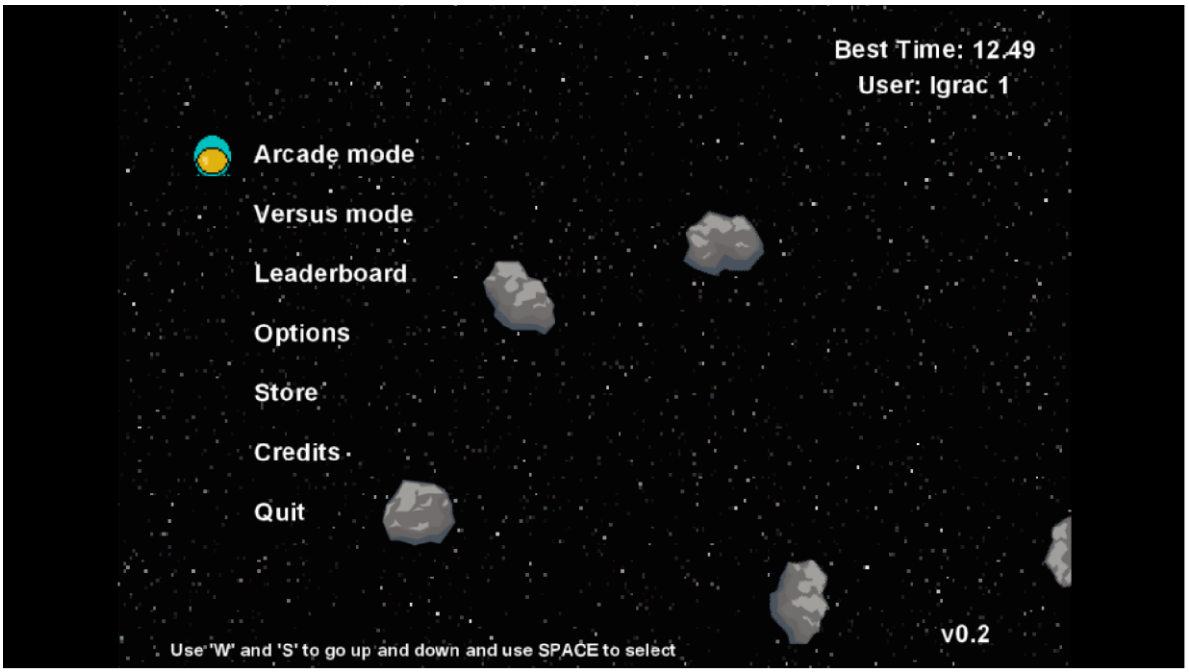
Slika 2. Koncept igre

Kako bi igrač imao kompetitivni duh. U igri se može vidjeti globalna lista najboljih vremena svih korisnika koji su poslali svoje najbolje vrijeme. Igrač može poslati svoje vrijeme na listu kada god je on zadovoljan svojim vremenom. Lista prikazuje top 10 igrača s njihovim vremenom. Kada igrač pošalje svoje vrijeme, lista se odmah ažurira.

Kako bi zadržali igrače što duže, dodali smo i mogućnost igranja u dvoje. Ako se odabere taj način igranja, tada svaki igrač dobije svoje tipke na tipkovnici. Umjesto jednog igrača se na ekranu pojave dva astronauta. Oba astronauta su obojana drugim bojama i imaju vlastite crte koje predstavljaju pregrijavanje. U ovom načinu igranja s ne prati vrijeme već se prati rezultat između igrača. Onaj igrač koji ostane duže živ dobije bod, dok gubitnik ne dobije ništa.

Igrač ima pravo mijenjati izgled svoga astronauta ako u glavnom izborniku odabere izbornik za mijenjanje izgleda astronauta. Astronauti su iste veličine. Jedno što ih razlikuje je boja ogledala i odjela svemirskog odjela koje nose. Osim boje neka odjela imaju poseban uzorak.

Igrač ima pravo i mijenjati neke postavke igrice. Igrač može mijenjati volumen pozadinske glazbe i volumen specijalnih efekata. Također može mijenjati svoje ime, koje se prikazuje na listi najboljih vremena. Može se mijenjati lokacije crte za pregrijavanje. Izbor je ispod ili iznad astronauta. Ako igrač to želi, može se i resetirati najbolje vrijeme igrača.



Slika 3. Glavni izbornik

3.4. Primjeri sličnih igara

3.4.1. Flappy Bird

Flappy Bird je igra u kojoj igrač pokušava održati pticu što duže može na životu. Ptica zbog utjecaja gravitacije neprestano pada. Ptica prolazi između cijevi koje variraju u visini. Također ptica ne smije dotaknuti tlo. Ako ptica dotakne tlo ili cijevi, igra završava.[18] Klikom ptica dobiva određenu visinu. Kao da je na kratko poletjela. Igra je dvodimenzionalna arkadna igra. Nema narativnu priču, već broji broj cijevi koje je ptica uspješno prošla. Flappy Bird je igra kreirana za mobilne uređaje 2013. godine. Izuzetno je brzo dobila nevjerojatno veliku popularnost. Zbog velike količine kompetitivnosti, igra je postala ekstremno ovisna. Do te mjere da je ljudi, većinom mlađi uzrasti, nisu mogli prestati igrati. Nakon nekog vremena kreator igre je povukao igru s tržišta jer je uvidio štetnost i ovisnost koju igra stvara. Unatoč lošem završetku, ovo je odličan primjer kako jednostavna arkadna dvodimenzionalna igra može pridobit ogromnu količinu korisnika.

3.4.2. Undertale

Undertale je jedna od najcjenjenijih i nagrađivanih dvodimenzionalnih igara u bližoj povijesti. Dobila je BAFTA nagradu za najbolju priču, NAVGTR nagradu za najbolju originalnu igru uloga, originalnu glazbu i za dizajn igre.[19] Undertale je dvodimenzionalna igra koja nije arkadna, već je tipa igre uloga. Izrazito je dugačka i ima detaljnu i fenomenalno razrađenu narativnu priču. Postoji više razloga isticanja ove igre. Prvi je taj što je igra nagrađena s velikim brojem nagrada. U tržištu u kojem dominiraju trodimenzionalne igre, ovo dokazuje da se dvodimenzionalne mogu nositi s njima. Drugi razlog je taj što je ova igra kreirana u GameMaker engine-u. Istom engine-u u kojem je rađena igra za ovaj rad.

3.4.3. Usporedba s računalnom igrom Asteroids

Atari-na igra Astroids je, što se tiče koncepta, najsličnija mojoj igri. Asteroids je dvodimenzionalna arkadna igra u kojoj se kontrolira svemirski brod i pokušava uništiti i izbjeći asteroide. Ako usporedimo igre, možemo vidjeti da imaju puno toga zajedničkog, ali i nekih velikih razlika.

Osim koncepta asteroida i svemira, zajednička stvar je stvaranje asteroida koji se kreću prema igraču. Razlika je u tome što se u Asteroids ti asteroidi mogu uništiti na manje dijelove dok ne budu uništeni. U mojoj igri se ti asteroidi ne uništavaju. Što se tiče glavnog lika, u Asteroids svemirski brod se može rotirati lijevo, desno, kretati ravno i pucati projektile, dok u mojoj igrici se glavni lik može naginjati lijevo i desno (do određene granice), kretati se naprijed, ali ne može pucati. U Asterods igrač također može izaći iz prostora igre i stvoriti se na suprotnoj strani ekrana, dok u mojoj to rezultira krajem igre. U mojoj igri se dobije jedan život po igri, dok se u Asteroid dobije nekoliko, što rezultira u većoj napetosti kod igranja moje igre. Velika je i razlika u praćenju rezultata igre. Asteroids prati rezultate kao broj bodova koje je igrač skupio unutar svih života jedne igre. S druge strane u mojoj igri se bilježi vrijeme u sekundama. Kraj igre u obje igrice rezultira dodirom asteroida s glavnim likom, no u Asteroids postoji i šansa stvaranja vanzemaljske letjelice koja šalje svoje projektile prema igraču. Još jedna velika razlika je tempo same igre. Asteroids je mnogo sporija igra u kojoj ima dovoljno vremena promisliti i namjestiti letjelicu, dok je moja dinamičnija i zahtjeva odluke u djeliću sekunde.

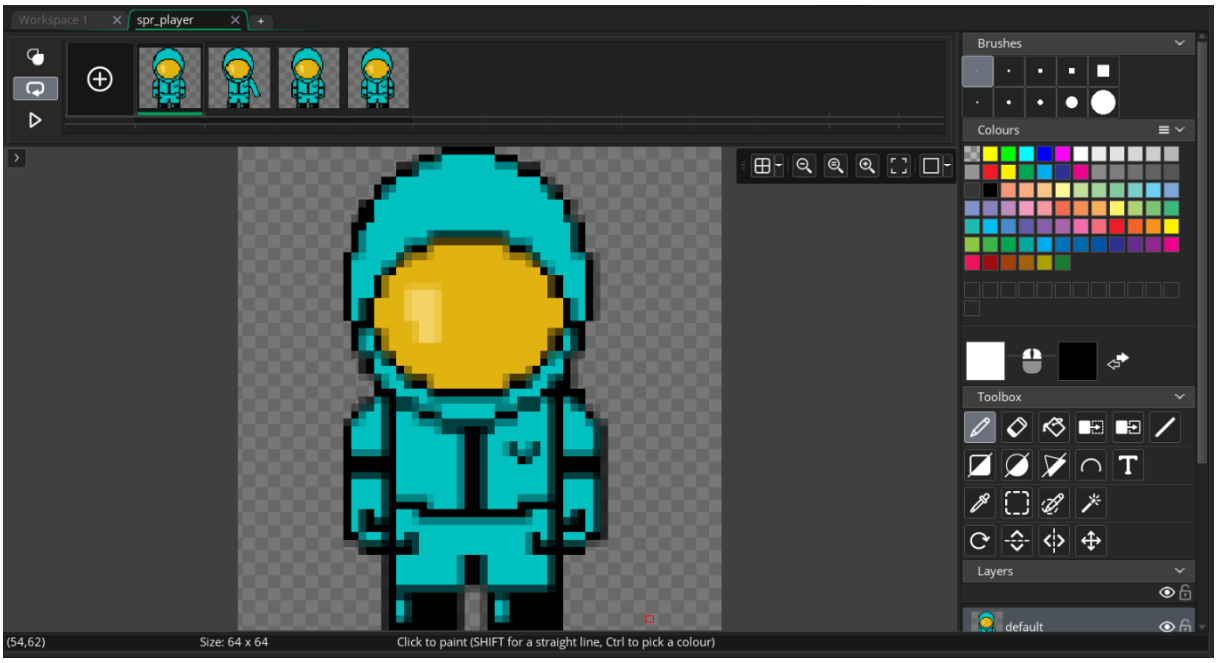
Kada se dvije igre usporede vide se mnoge sličnosti, ali i mnoge različitosti. Tematika je slična, ali velika većina funkcionalnosti načina igranja i načina bodovanja je različita.

4. Izrada grafičkih resursa

Za kreiranje grafičkih resursa postoje mnogi vanjski alati. Oni daju velike mogućnosti i različite kvalitete. Te alate je naravno moguće koristiti i naknadno uvesti u GameMaker, ali ja to nisam radio. Sam GameMaker Studio daje mogućnosti uređivanja i izrade grafičkih resursa. Pokretanje te opcije izvodimo klikom na postojeći grafički resurs ili desnim klikom i odabirom „create->sprite“.

Slika 4. Sučelje grafičkog resursa

Grafički izbornik nam daje mogućnost kreiranja jedne slike ili cijele animacije, postavljanja veličine slike i centralne točke slike i masku kolizije slike. Na primjeru vidimo karikaturu astronauta. U ovom slučaju smo otvorili postojeći resurs. Taj resurs se može uređivati klikom na gumb „Edit Image“.



Slika 5. Sučelje uređivanja grafičkog resursa

Otvara nam se nova kartica u kojoj nam se prikazuje platno slike i svi alati dostupni unutar GameMaker-a. Imamo mogućnost kreiranja slojeva i posebno ih uređivati, crtati slobodnom rukom, crtati geometrijske oblike, mijenjati boju, označavati i micati dijelove slike, brisati i pisati tekstove. U slučaju kreiranja animacija, imamo mogućnost kreiranja koraka animacije i prolaska kroz animaciju u istom okviru.

5. Funkcionalnosti

5.1. Kretanje igrača

Najbitnija funkcionalnost ove igre je kretanje astronauta koji predstavlja glavnog lika. Bez ove funkcionalnosti igra ne bi imala smisla. Rezultat bi bio nasumičan, ovisan o sreći, i igraču ne bi bilo potrebno učiti mogućnosti, limite i pravovremeno reagiranje. Sav smisao igre bi bio uništen.

Za kretanje igrača nam je potreban jedan objekt i dvije skripte. Objekt igrača smo nazvali obj_player. Postavke tog objekta su: vidljiv je, ne koristi fiziku, nije stalan objekt, ne ponaša se kao solidan objekt i maska objekta je jednaka maski grafičkog prikaza objekta. Objekt igrača ima pet događaja s kojima obavlja svoju funkcionalnost.

U objektu Kreiranja deklariramo i inicijaliziramo početne varijable. Kod događaja kreiranja izgleda ovako:

```
gas = 20;
maxgas = 30;
grav = 6;
sprite_index=obj_score.sprite;
image_angle=0;
```

Događaj je mali, u smislu linija koda, ali je izrazito važan jer nam postavlja važne varijable. Varijabla „gas“ je količina, ili je bolje rečeno temperatura, koju sadrži način kretanja. Varijabla „maxgas“ predstavlja maksimalnu količinu kretanja koju igrač može izvesti bez da se počne pregrijavati. Varijabla „grav“ predstavlja količinu gravitacije koja vuče igraća prema dnu ekrana. Varijabla „sprite_indeks“ je varijabla u koju spremamo sliku koja predstavlja igrača. Slika igrača se može mijenjati i to ćemo razjasniti u daljnjim poglavljima. Slika se dobiva iz varijable objekta „obj_score“. Taj objekt je stalan objekt i prisutan u svim dijelovima igre. Na kraju imamo varijablu „image_angle“ koja nam postavlja kut u kojem slika igrača gleda. Taj kut koristimo za kretanje.

Najvažniji događaj u objektu igrača je događaj koraka. On se višestruko poziva po sekundi, ovisno o postavkama igre. Kod događaja koraka:

```
script_execute(scr_input);
script_execute(scr_move);
image_index=0;
```

Radi skraćenosti i urednosti koda igrača, većinu njegovog koda smo premjestili u skripte. Jedine naredbe koje su ostale su naredba za pozivanje i izvršavanje skripte (script_execute()), te naredba koja postavlja indeks slike. Indeks slike se koristi kod animacija. Svaki indeks je jedan okvir u procesu animiranja. Kod skripti input i skripti move izgleda ovako:

```
function scr_input() {
      up_key = max(keyboard_check(ord("W")),keyboard_check(vk_up));
      left_key =max( keyboard_check(ord("A")),keyboard_check(vk_left));
      right_key = max(keyboard_check(ord("D")),keyboard_check(vk_right));
      down_key = max(keyboard_check(ord("S")),keyboard_check(vk_down));
      action_key = keyboard_check_pressed(vk_enter);
      start_key = keyboard_check_pressed(vk_space);
      menu_up_key=
max(keyboard_check_pressed(ord("W")),keyboard_check_pressed(vk_up));
      menu_left_key                                                    =max(
keyboard_check_pressed(ord("A")),keyboard_check_pressed(vk_left));
      menu_right_key=
max(keyboard_check_pressed(ord("D")),keyboard_check_pressed(vk_right));
      menu_down_key=
max(keyboard_check_pressed(ord("S")),keyboard_check_pressed(vk_down));
}

function scr_move() {
      if(left_key == 1 && image_angle < 40)
          image_angle += 2;
      else if(image_angle > 0 && left_key != 1)
          image_angle -= 4;
      if(right_key == 1 && image_angle > -40)
              image_angle -=2;
      else if(image_angle < 0 && right_key != 1)
          image_angle += 4;
      // Boost
      if(up_key == 1 && alarm[0]<= 0 && gas > 0)                    // Go up
      {
          speed = 10;
          grav = 0;
          direction = 90+image_angle;
          gas -=1;
      }else{
          if(gas ==0)  alarm[0]=room_speed;
          if(gas < maxgas)  gas += .5;
          gas=ceil(gas);
          speed = 6;
          if(image_angle > 30)
      direction = 230;
          else if(image_angle <= 30 && image_angle > 20)
              direction = 240;
          else if(image_angle <= 20 && image_angle > 10)
```

```
            direction = 250;
        else if(image_angle <= 10 && image_angle > 5)
            direction = 260;
        else if(image_angle <= 5 && image_angle > -5)
            direction = 270;
        else if(image_angle <= -5 && image_angle > -10)
            direction = 280;
        else if(image_angle <= -10 && image_angle > -20)
            direction = 290;
        else if(image_angle <= -20 && image_angle > -30)
            direction = 300;
        else if(image_angle <=-30)
            direction = 310;
    }
}
```

Skripta input, nazvana scr_input, kreira varijable za gore, dolje, lijevo, desno, izbornik gore, izbornik dolje, izbornik lijevo, izbornik desno i varijable za start i akciju. Vrijednost tih varijabli je 1 ili 0. To smo postigli korištenjem dvije naredbe. Prva je „keyboard_check()“ koja vrača 1 ako je pritisnuta tipka koja odgovara kodu unutar zagrada naredbe ili 0 ako nije pritisnuta. Druga naredba je „max()“. Ona vrača maksimalnu vrijednost odabranu iz varijabli između zagrada. Ona nam je potrebna jer dajemo opciju korištenja više tipki za istu funkciju. Naredba „ord()“ vraća Unicode vrijednost znaka u zagradi.

Skripta move, nazvana scr_move, se koristi za promjenu pozicije igrača. Prva stvar koju skripta radi je provjere dali je pritisnuta tipka za naginjanje i dali je dosegnut maksimalni kut za naginjanje u tom smjeru. U slučaju da je moguće naginjanje, pomoću varijable „image_angle“ se slika astronauta rotira u odabranom smjeru. Nakon toga se provjerava da li je pritisnuta tipka za kretanje, da li imamo dovoljno resursa i da li je alarm ne aktivan. U tom slučaju negiramo gravitaciju i varijablom „speed“ postavimo brzinu kretanja i varijablom „direction“ pokrenemo kretanje u smjeru kuta slike astronauta. Dok god pritišćemo tipku za kretanje se troši resurs kretanja. U slučaju da zahtjevi provjere nisu zadovoljeni, kreiramo kod za padanje igrača uzrokovano gravitacijom. Ako nemamo resursa za kretanje postavljamo alarm koji služi kao hlađenje resursa i čekamo dok se resurs u potpunosti ne vrati. Ako je resurs manji od njegovog maksimuma, on se postupno puni. Ostatak koda u skripti služi za pomicanje igrača dok pada. Promjenom kuta pomičemo putanju padanja igrača. Time dobivamo efekt kontroliranog padanja.

Objekt igrača kao događaj ima postavljen jedan alarm. U tom događaju ne postoji nikakav kod. Alarm služi kako bi skripta za kretanje imalo mogućnost koristiti ga. S tim alarmom dobijemo efekt pregrijavanja korisnikovog pogona.

Zadnja dva događaja su događaj sudara s objektom asteroida i događaj izlaska izvan sobe. Oba događaja imaju jednak kod. Unutar događaja je jedna naredba. Ta naredba je „instance_destroy()“. Ona uništava objekt igrača i drugi objekti znaju da trebaju aktivirati kraj igre.

5.2. Prikaz pregrijavanja mlaznog pogona

Prikaz pregrijavanja pogona igrača se izvršava u objektu zvanom obj_boost_bar. Taj objekt sadrži tri događaja, a oni su: događaj kreiranja, događaj koraka i događaj crtanja. Objekt nema početni grafički prikaz i nije stalan, ali je vidljiv.

Kod događaja kreiranja glasi:

```
boost=30 - obj_player.gas;
max_boost=30;
boost_height = 5;
boost_width = 86;
boost_x = x;
boost_y = y;
boost_offset_x = 0;
boost_offset_y = 0;
if(obj_score.bar_option == 0){
      boost_offset_x = -40;
      boost_offset_y = 60;
}
if(obj_score.bar_option == 1){
      boost_offset_x = -40;
      boost_offset_y = -60;
}
sprite = spr_green_bar;
reset=false;
```

Varijabla „boost“ predstavlja količinu linije koja se treba popuniti. Taj iznos dobivamo iz objekta igrač. Sljedeće varijable su vezane za pomak lokacije linije. Pošto u opcijama imamo više načina prikaza linije, a to su iznad ili ispod astronauta, moramo provjeriti u objektu „obj_score“ koja je vrijednost varijable „bar_option“. Na kraju u varijablu „sprite“ postavimo grafički resurs koji predstavlja zelenu boju. Postoje 4 različita resursa koji predstavljaju različite stadije pregrijavanja. S njima linija može biti zelena, žuta, narančasta ili crvena ako se pregrije.

Kod događaja koraka za objekt prikaza pregrijavanja zgleda ovako:

```
boost=30 - obj_player.gas;
if(boost >= 30) reset=true;
if(boost <= 3) reset=false;
boost_x= obj_player.x + boost_offset_x;
boost_y= obj_player.y + boost_offset_y;
if(reset == false and boost < 30) sprite=spr_orange_bar;
if(reset == false and boost < 25) sprite=spr_yellow_bar;
if(reset == false and boost < 15) sprite=spr_green_bar;
if(reset == true) sprite=spr_red_bar;
```

Svakim korakom se ažurira količina crte. Na način da se ponovno dobije podatak iz objekta igrač. Zatim se provjeri da li se treba postaviti pregrijavanje u varijabli „reset“. U svakom

koraku se ispravlja pozicija na kojoj će se crtati linija pregrijavanja. Na kraju se provjerava količina crte i odabire grafički element koju će je predstavljati.

U događaju za crtanje imamo dvije funkcije. Te funkcije su „draw_sprite_stretched(sprite,0,boost_x,boost_y,(boost/max_boost)*boost_width,boost_height)“ i „draw_sprite(spr_boost_frame,0,boost_x,boost_y)“. Prva funkcija uzima grafički resurs, postavlja mu indeks, i crta ga na zadanim koordinatama. Još jedna stvar koja funkcija radi je ta da grafički resurs rasteže onoliko koliko je zadao u zadnja dva argumenta. Druga funkcija crta okvir linije pregrijavanja na istim koordinatama kao i sama linija.

5.3. Asteroidi

Asteroidi se izvršavaju korištenjem dva objekta. Prvi objekt se naziva obj_asteroid i njegova uloga je predstavljati asteroid koji se pokušava sudariti s igračem. Drugi objekt se naziva obj_spawner i njegova uloga je kreiranje asteroida u određenim vremenskim periodima.

Objekt stvaratelj ima događaj pokretanja sobe i četiri alarma. Događaj pokretanja sobe provjerava da li objekt igrač postoji. To provjerava s „instance_exists(obj_player)“. Funkcija vrača istinu ako igrač postoji unutar ove sobe. Nakon toga postavlja alarme na tri sekunde. To izvodimo tako da upišemo „alarm[0] = room_speed*3“. Svaki alarm ima sličan kod. Jedina razlika je koordinate na kojima kreiraju objekt asteroid. Unutar alarma prvo se provjerava da li igrač postoji, nakon toga se u lokalnu varijablu „time“ generira vrijeme nakon kojeg će se ponovno generirati asteroid. Poslije postavljanja vremena generiramo novu instancu objekta asteroida funkcijom „instance_create()“. Nakon toga ponovno postavimo alarm s generiranim vremenom. Kod alarma glasi:

```
if(instance_exists(obj_player)){
var time = max(irandom(3),1);
  instance_create(0,0,obj_asteroid);
    alarm[0]=room_speed*time;}
```

Objekt asteroid sadrži tri događaja. To su događaj kreiranja, događaj koraka i događaj izvan sobe. Događaj izvan sobe je iznimno jednostavan. Kreira se lokalna varijabla „chance“ u koju se nasumično generira broj između 0 i 5. U slučaju da je generirani broj različit 1, smjer asteroida se mijenja za 180 stupnjeva i vrača u vidljivi okvir. U slučaju da je generirani broj jednak 1, onda se instanca asteroida uništava.

Kod za događaj kreiranja asteroida glasi:

```
image_index = irandom(3);
image_speed = 0;
rotation = irandom(20);
rot_dir = irandom(2);
speed = 6;
direction = irandom(360);
```

Grafički model za asteroide ima u sebi četiri slike različitih asteroida. Svaka slika se nalazi na svom indeksu. Indeksirane slike se tipično koriste za animacije, ali u ovom slučaju koristimo za različite oblike asteroida. Taj indeks spremamo u varijablu „image_index“. Varijabla „image_speed“ je brzina promjene animacije. Pošto se ne radi o animaciji, ona mora biti 0. Nadalje generiramo brzinu rotacije i smjer rotacije asteroida. Na kraju imamo varijable za brzinu, koja je unaprijed zadana, i za smjer koji je nasumičan. U događaju koraka, ovisno o smjeru rotacije, mijenjamo kut slike objekta. To izvodima na način da zbrajamo ili oduzimamo kut koji je spremljen u varijablu „rotation“.

5.4. Praćenje rekorda i trenutnog vremena

Za praćenje rekorda i vremena igre su nam zadužena dva objekta. Ta dva objekta su obj_timer i obj_score. Ako pogledamo cjelokupnu igru, obj_score je možda najvažniji objekt. On je jedan od jedinih stalnih objekata što znači da je on prisutan u svakoj sobi. Njega koristimo kao most podataka između objekata koji, zbog toga što se nalaze u različitim sobama, ne mogu komunicirati. Objekt score prati trenutačno vrijeme igrača, vrijeme s kojim je igrač završio, najbolje vrijeme igrača, opcije gdje prikazati liniju pregrijavanja, korisničko ime igrača i sliku koja predstavlja igrača. Objekt score također kontrolira kraj igre i vračanje na glavni izbornik. To izvodi u događaju koraka. Isto tako u tom događaju pušta zvuk koji označava smrt astronauta i bilježi krajnji rezultat igrača.

Kod događaja koraka u objektu score glasi:

```
if(room==rm_arcade){
      time =obj_timer.timer ;
      if(!instance_exists(obj_player)){
        audio_play_sound(snd_death,10,false);
        room_goto(rm_endgame);
      }
}
if (room==rm_endgame)
    new_time=time;
if(new_time > time1)
      time1 = new_time;
```

Kako bismo znali kada trebamo mjeriti vrijeme, a kada zaustaviti igru, moramo znati u kojoj prostoriji se nalazi objekt score. Svaki objekt ima varijablu „room“ koja u sebi ima upisanu trenutačnu sobu. U slučaju da je objekt u sobi „rm_arcade“, onda čitamo vrijeme iz objekta timer-a. Ako smo unutar te sobe i nema objekta igrača, to znači da je igrač izgubio igru. U tom slučaju je potrebno pustiti zvuk korištenjem „audio_play_sound()“ funkcije. Zvuk kojeg puštamo se naziva „snd_death“. Zatim izvršimo promjenu sobe naredbom „room_goto()“. Soba „rm_endgame“ je soba u kojoj se prikazuje trenutačan i najbolji rezultat i nakon par sekundi vrača na glavni izbornik. Kada se nalazimo u toj sobi, postavimo da je zadnje zabilježeno vrijeme novo vrijeme. Ako je novo vrijeme bolje od prijašnjeg najboljeg vremena, onda novo vrijeme postaje najbolje vrijeme.

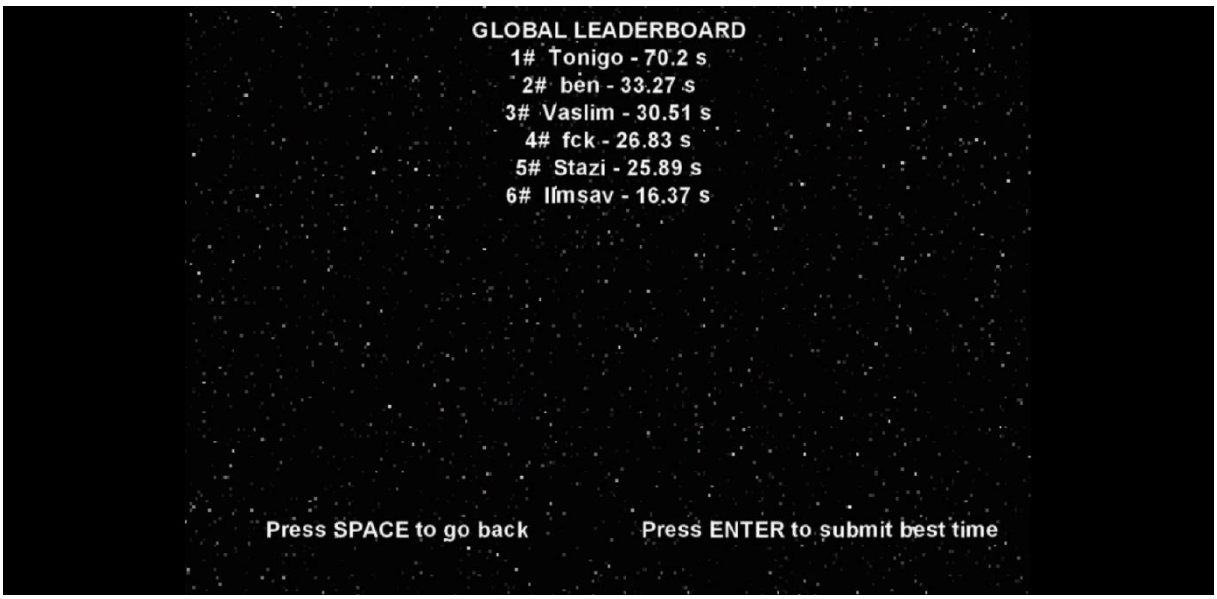
Objekt timer ima događaje kreiranja, događaj koraka i događaj crtanja. U događaju kreiranja samo deklarira varijablu „timer“ i postavlja početnu vrijednost na „0.0“. U događaju koraka imao par linija koda koji nam služi za povećavanje vrijednosti u varijabli „timer“. To postižemo linijom koda koja glasi: „timer = timer + delta_time/1000000;“. Nakon što povećamo vrijeme, to vrijeme trebamo ispisati igraču. Ispisivanje vremena izvodimo u događaju crtanja.
U događaju crtanja prvo namjestimo postavke fonta kojim ćemo ispisati rezultat. Font „fnt_menu“ je font koji smo unesli u GameMaker Studio iz vanjskih izvora. Nakon izbora fonta, boje i centriranja, ispišemo tekst pomoću funkcije „draw_text()“. Tekst se smatra novim redom, pa koristimo funkciju „string_hash_to_newline()“. Pošto varijabla „timer“ nije tipa tekst, moramo je pretvoriti u tekst. To izvodimo uporabom funkcije „string()“. Kod događaja crtanja glasi:

```
draw_set_halign(fa_center);
draw_set_valign(fa_middle);
draw_set_font(fnt_menu);
draw_set_colour(c_white);

draw_text(x,y,string_hash_to_newline("Time: "+string(timer)));
```

5.5. Globalna lista rezultata

Kako bi napravili globalnu listu rezultata, potrebna nam je jedna točka iz koje sve verzije igre čitaju. Ta točka je apache poslužitelj. Kreirali smo globalnu listu rezultata kako bi napravili igru kompetitivnijom. Umjesto da se igrač bori protiv svojeg najboljeg rezultata, on se bori protiv ostalih igrača.



Slika 6. Lista najboljih rezultata

Listu rezultata spremamo u MySQL bazu podataka. U bazi podataka imamo jednu relaciju u kojoj zapisujemo korisničko ime i vrijeme igrača koji koristi to korisničko ime. Kako bi dobili komunikaciju između poslužitelja i aplikacije, moramo kreirati skripte koje se spremaju na poslužitelju i izvršavaju neku zadaću kada su pozvane. U našem slučaju koristimo skripte napisane u PHP programskom jeziku. Imamo dvije skripte. Prva je namijenjena za čitanje i naziva se „ispis.php“. Druga skripta je namijenjena za unos novog podatka u bazu podataka. Ta skripta se naziva „unos.php“. Kodovi skripte ispis i unos glase:

```
<?php //SKRIPTA ISPIS
$conn = mysqli_connect($servername, $username, $password);
if (!$conn) {
  die("Connection failed: " . mysqli_connect_error());
}
mysqli_select_db($conn,$dbname);
$sql = "SELECT username, time FROM leaderboard order by time DESC LIMIT 10";
$result = mysqli_query($conn, $sql);
$response="";
$br=1;
if (mysqli_num_rows($result) > 0) {
  while($row = mysqli_fetch_assoc($result)) {
        $response.= $br."#  ".$row["username"]." - ".$row["time"]." s\n";
        $br++;
  }
  echo $response;
} else {
  echo "0 results";
}
mysqli_close($conn);
?>

<?php //SKRIPTA UNOS
$name = $_POST["username"];
$time = $_POST["time"];
$conn = mysqli_connect($servername, $username, $password);
if (!$conn) {
  die("Connection failed: " . mysqli_connect_error());
}
mysqli_select_db($conn,$dbname);
$sql = "SELECT * FROM leaderboard WHERE username = '".$name."'";
$result = mysqli_query($conn, $sql);
if (mysqli_num_rows($result) > 0) {
    $row = mysqli_fetch_assoc($result);
    $oldTime = (float) $row["time"];
    $newTime = (float) $time;
    if($newTime > $oldTime){
        $sql = "UPDATE leaderboard SET time = ".$time." WHERE username =
'".$name."'";
    }
} else {
    $sql = "INSERT INTO leaderboard (username, time) VALUES ('".$name."',
".$time.")";
}
$result = mysqli_query($conn, $sql);

mysqli_close($conn);
?>
```

Kada aplikacija pošalje pozivanje skripte ispis. Ona se asinkrono izvršava i vrača listu deset najboljih igrača i njihova vremena. Unutar skripte se prvo trebamo spojiti na bazu podataka. Za to koristimo funkciju „mysqli_connect()“. U varijablama, koje su proslijeđene kao argumenti, se nalaze podaci o imenu servera, lozinki i korisniku. Koristimo MySQLi proceduralnu metodu komunikacije s bazom podataka. Nakon spajanja i provjere stanja veze pomoću „mysqli_select_db()“ odaberemo traženu relaciju. Kreiramo SQL upit koji nam čita,

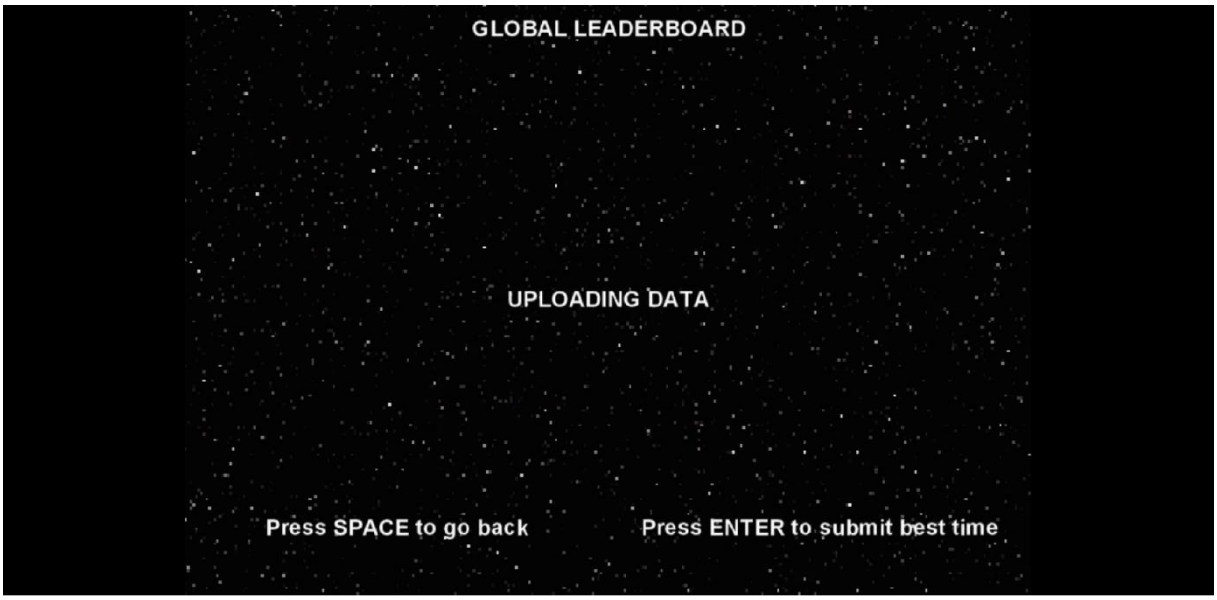
sortira i uzima najboljih 10 zapisa. Taj upit pomoću „mysqli_query()“ izvršimo nad bazom i dobijemo rezultat. U slučaju da rezultat ima više od nula redova, nadodajemo ga u varijablu koja nam služi kao odgovor za aplikaciju. Kreirali smo i pomoćnu varijablu koja nam služi za predstavljanje brojeva koji označavaju poziciju igrača u listi. Svaki zapis baze podataka pretvaramo u asocijativni niz pomoću „mysqli_fetch_assoc()“. Nakon svih iteracija vračamo aplikaciji tekstualni podatak koji predstavlja listu ili u slučaju da je lista prazna vračamo tekst u kojem piše da ima nula rezultata. Pomoću „mysqli_close()“ od spajamo se od baze podataka.

Aplikaciji skripti unos šalje dva podatka pomoći POST metode. Te podatke spremamo u varijable pomoću super globalnih varijabli „$_POST“. U skripti unos se spajamo na bazu podataka na isti način kao i u skripti ispis. Kada smo se uspješno spojili na bazu, kreiramo i izvršimo upit s kojim provjeravamo da li korisnik već ima zapisan rezultat na svoje korisničko ime. U slučaju da ima već zapisani rezultat, onda čitamo stare zapise i provjeravamo da li je novi rezultat bolji od starog. U slučaju da je novi bolji, ažuriramo zapis u bazi podataka i korisničkom imenu postavimo bolji rezultat. Ovo smo napravili kako se baza podataka ne natrpa s velikom količinom podataka, a od te velike količine koristimo samo mali broj. Na ovaj način smo smanjili količinu zauzete memorije i povećali brzinu odgovora baze. Ako igrač nema već kreiran zapis s korisničkim imenom, onda se kreira novi zapis u bazi podataka.

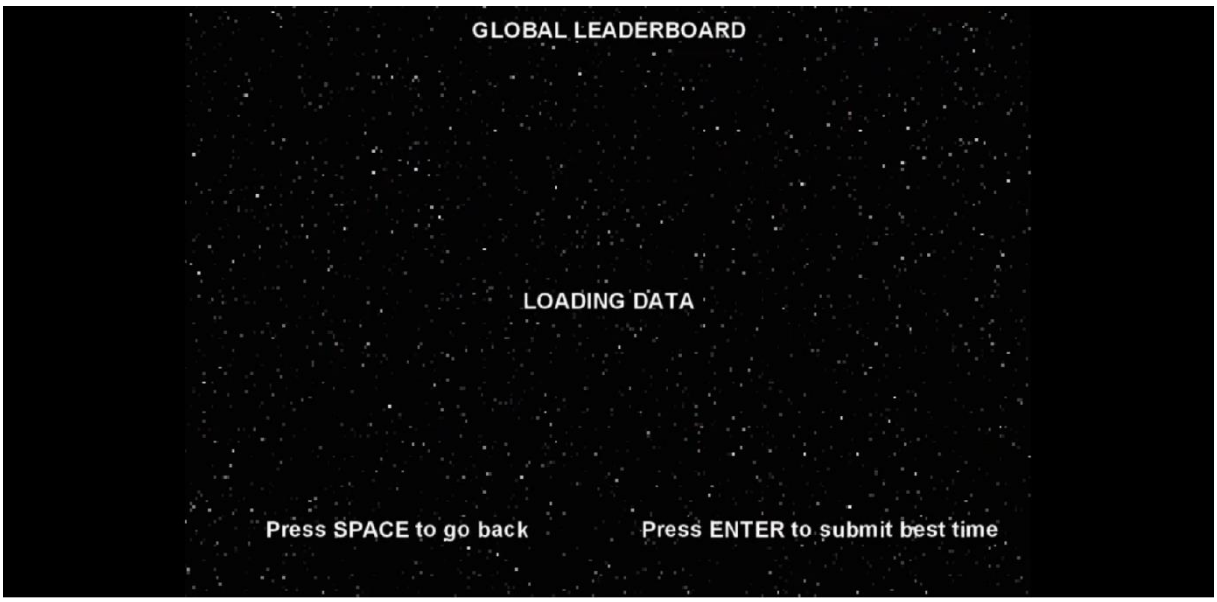
Sa strane igre, za komunikacijom s poslužiteljem koristimo samo jedan objekt. Taj objekt smo nazvali „obj_leaderboard“. On u sebi sadrži sedam događaja. Ti događaju su: događaj kreiranja, alarm, događaj crtanja, dva događaja pritiska tipke, događaj pokretanja sobe i događaj primanja asinkronih HTTP poruka.

U događaju kreiranja kreiramo varijable instance za daljnje funkcionalnosti objekta i dodjeljujemo im početne vrijednosti. Svim varijablama je dodijeljena početna vrijednost „false“, osim varijablama „url“, „url_post“ i „r_string“. Varijabli „r_string“ smo zadali prazan tekstualni podatak. Tu varijablu koristimo za spremanje povratnih informacija s poslužitelja. U varijable „url“ i „url_post“ smo spremili URL adresu koja poziva PHP skripte unos i ispis.

Događaj crtanja ispisuje tekst na ekranu ovisno o stadiju u kojem objekt. Ako je objekt dohvatio listu, on ju ispisuje na ekranu. U slučaju da je još uvijek u stanju dohvaćanja, ispisuje poruku o dohvaćanju. Kada se pritisne gumb za slanje svoga rezultata, ispiše se poruka slanja dok se podatak ne pošalje.



Slika 7. Slanje rezultata



Slika 8. Dohvaćanje rezultata

Događaj početka sobe pokreče dohvaćanje liste rezultata. U ovom događaju se nalazi jedna instrukcija. Ona je „http_get()“. Ta funkcija šalje signal metodom GET prema skripti na adresi koja se upisuje kao argument funkcije.

Događaj pritiska na tipku pokreče slanje rezultata prema poslužitelju. Kod unutar događaja prvo postavi varijablu stanja u „true“. Zatim dohvati korisničko ime igrača iz objekta score i iz toga imena makne navodnike koristeći funkciju „string_replace_all()“. Razlog micanja je taj da u slučaju navodnika dolazi do pogreška u PHP skriptama i ne zapisivanja u bazu podataka. Nakon toga se pomoću „http_post_string()“ funkcije šalju podaci POST metodom prema poslužitelju. Argumenti funkcije su URL adresa skripte i tekstualni podatak kojeg smo

unaprijed pripremili. Na kraju postavimo alarm koji će ponovno pozvati ispis liste. Kod događaja glasi:

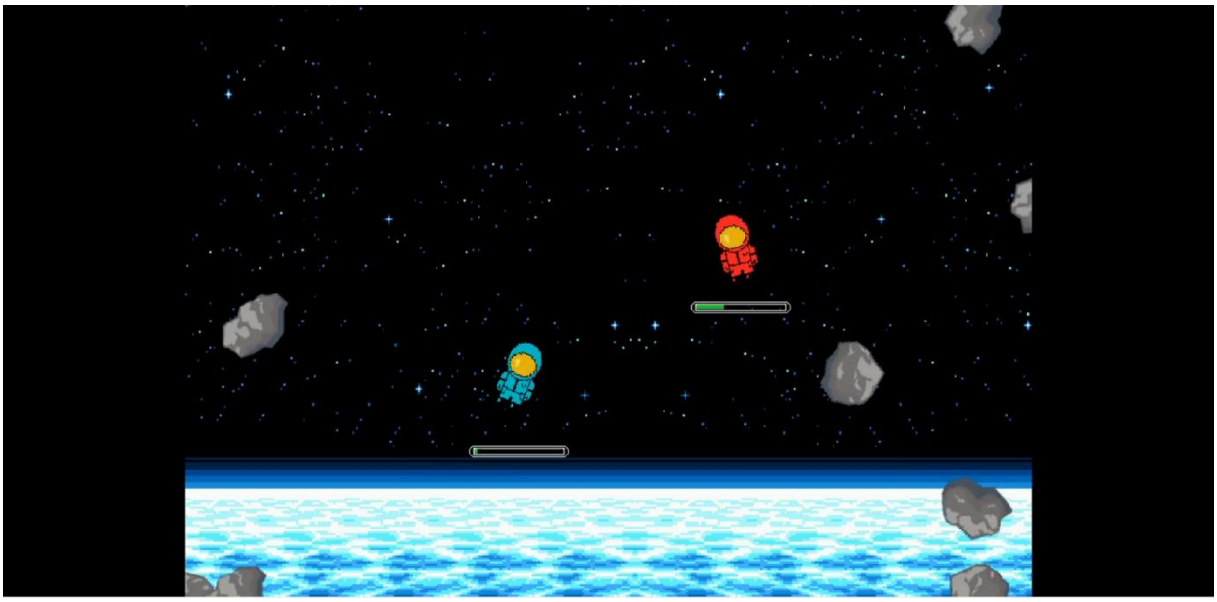
```
sending = true;
var username_post=obj_score.username;
username_post = string_replace_all(username_post,"'","");
var str = "username=" + username_post + "&time=" + string(obj_score.time1);
post = http_post_string(url_post, str);
alarm[0]=room_speed*2;
```

Događaj asinkronog HTTP signala se aktivira kada poslužitelj pošalje HTTP odgovor. Sam događaj prvo provjeri da li identifikator odgovora odgovara GET metodi. Ako je HTTP zahtjev uspješno obrađen, njegov status bi trebao biti 0. Ako dođe neka druga vrijednost, postavljamo varijablu pogreške u „true“ i ispisujemo pogrešku u događaju crtanja. No, ako se sve pravilno izvrši, spremamo odgovor poslužitelja unutar predefinirane varijeble i postavimo varijablu završetka dohvaćanja u „true“. Odgovor servera se nalazu u takozvanoj DS mapi. DS mapa je struktura u GameMaker Studiju koja sprema podatke kao ključ i vrijednost. Za dolazak do vrijednosti koristimo funkciju „ds_map_find_value()“. Kao argument koristimo ključ. Kod događaja glasi:

```
if(ds_map_find_value(async_load, "id") == get){
      if(ds_map_find_value(async_load, "status") == 0){
            r_string = ds_map_find_value(async_load, "result");
            loaded=true;
      }else{
            error=true;
      }
}
```

5.6. Igranje u dvoje

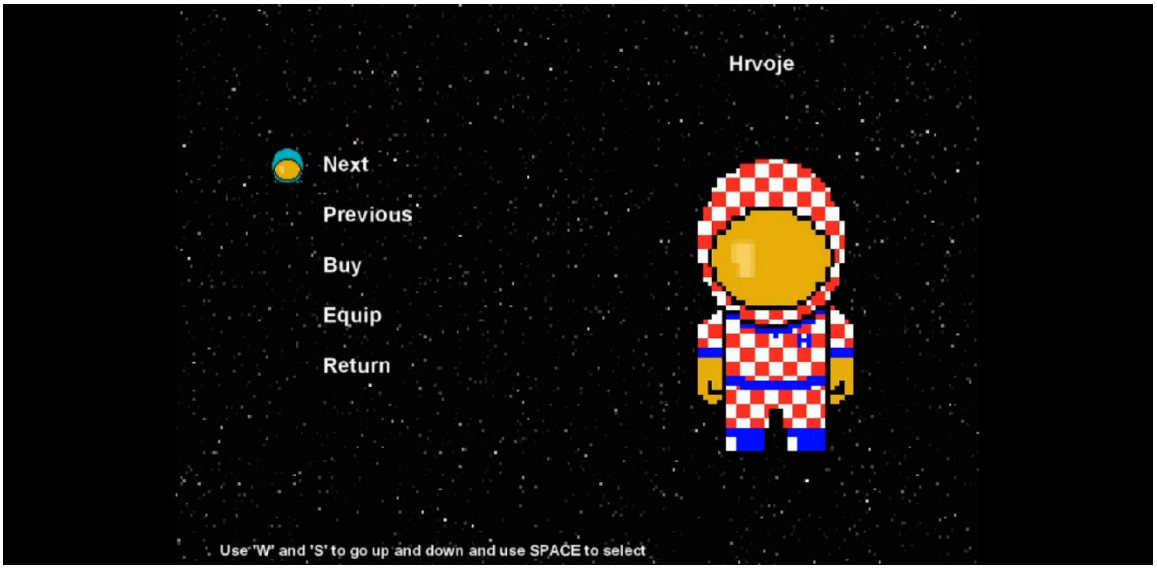
Igranje u dvoje smo jako lagano postigli jer već imamo objekte koji većinu zahtjeva ispunjavaju. Objektima poput objekta igrača i objekta linije pregrijavanja smo kreirali objekte djecu koji nasljeđuju sve događaje i postavke, ali smo određene skripte modificirali. Za svakog igrača smo kreirali novi par objekta igrač i linija pregrijavanja. Stvorili smo novu sobu u koju smo postavili sve potrebne nove objekte. Za upravljanje svakog igrača smo modificirali skriptu „move“ na način da jedan igrač upravlja s tipkama W,A,S,D, dok drugi igrač upravlja svojim likom sa strelicama. Kako ne bi bilo zabune, astronaut svakog igrača je obojan drugačijom bojom. Astronaut prvog igrača je obojan u plavu, dok je astronaut drugog igrača obojan u crvenu boju.



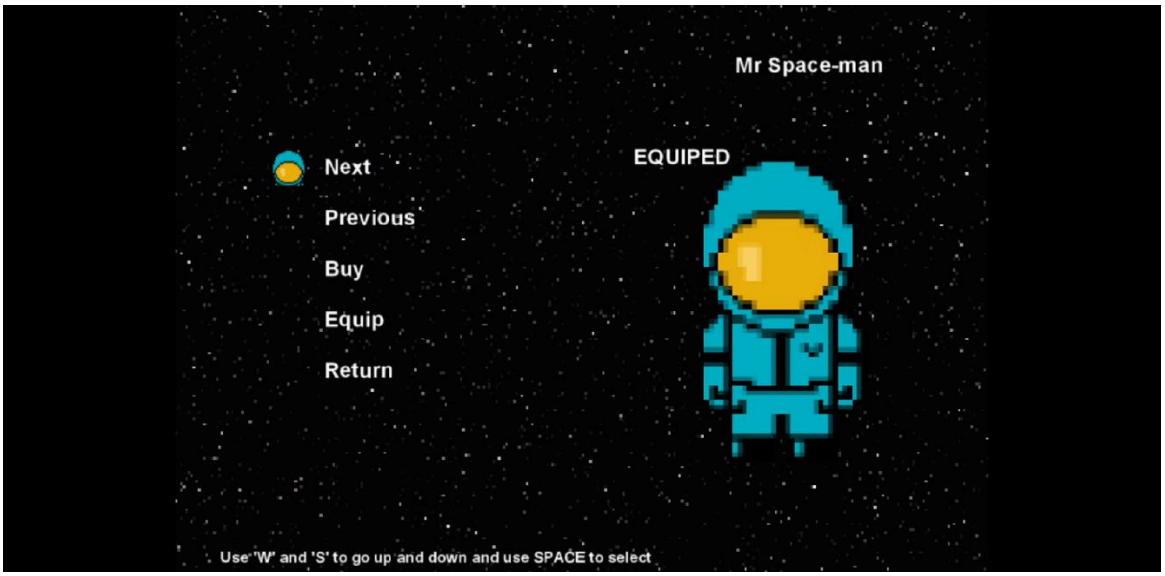
Slika 9. Igranje u dvoje

5.7. Promjena izgleda lika

Igra daje mogućnost promijene izgleda astronauta. U glavnom izborniku postoji podizbornik koji daje mogućnost pregleda svih dostupnih izgleda i odabir željenog. Kada se željeni odabere, mijenja se varijabla u objektu „score“ koja pokazuje objektu igrač koji izgled treba prikazati. U isto vrijeme se ažuriraju spremljeni podaci na računalu igrača, ali o tip podacima ćemo u pričati u nastavku rada.



Slika 10. Pregled izgleda astronauta

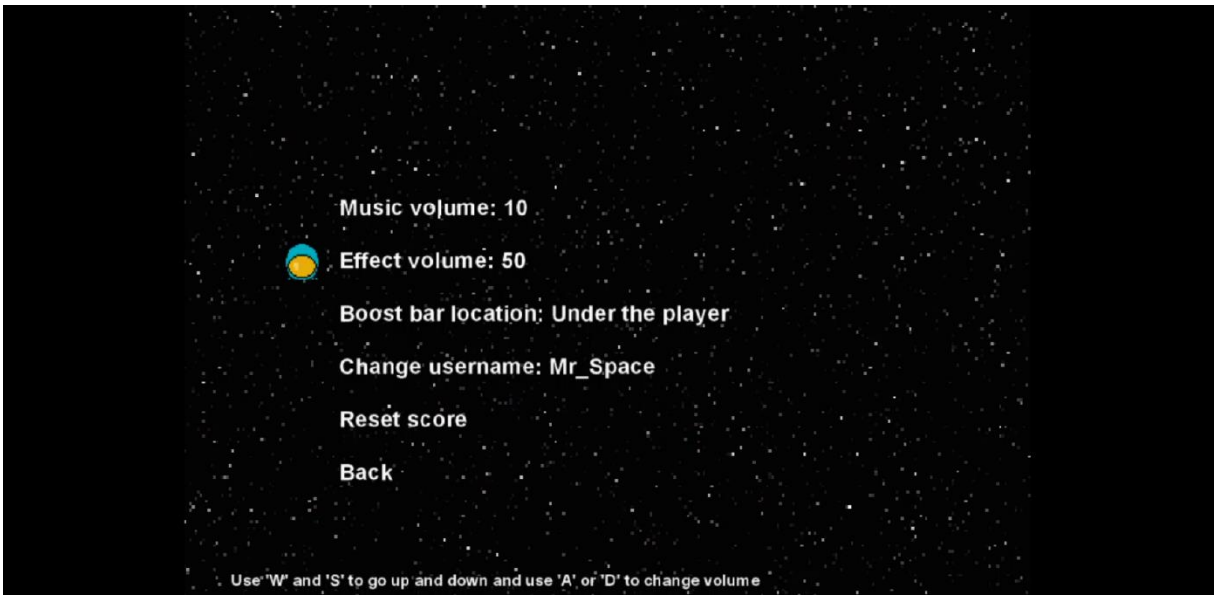


Slika 11. Odabir izgleda astronauta

5.8. Spremanje i čitanje napretka

Za spremanje i čitanje napretka smo kreirali tri skripte koje kreiraju i čitaju tri datoteke spremljene na računalu igrača. Tri datoteke spremaju različite podatke u sebi. Prva lokalno sprema najbolji rezultat igrača. Ta datoteka se najviše čita i ažurira. Ažurira se nakon svake igre koju igrač odigra, a čita se pri uključivanju igre.

Druga skripta sprema opcije koje je igrač postavio. Opcije koje se mogu spremati su: volumen glazbe, volumen efekata, pozicija linije pregrijavanja i korisničko ime igrača. Ta datoteka se čita pri pokretanju igre i ažurira se izlaskom iz izbornika s opcijama.



Slika 12. Izbornik opcija

Treća datoteka sadrži podatke o izgledu astronauta koje je igrač odabrao. Spremaju se svi indeksi koje imaju u igri i indeks koji je trenutačno odabran. Datoteka se ažurira izlaskom iz izbornika za promjenu izgleda, a čita se pri pokretanju igre.

Za primjer spremanja i čitanja smo odabrali prvu datoteku koja sadrži najbolje vrijeme igrača. Skripta ima dvije korisničke funkcije. Prva je „scr_save()“, a druga je „scr_load()“. Tijekom spremanja prvo provjerimo da li datoteka već postoji. U slučaju da postoji, brišemo ju. Nakon toga otvorimo novu datoteku s funkcijom „file_text_open_write()“, dohvatimo podatak iz objekta „score“ i zapišemo taj realan broj u datoteku pomoću „file_text_write_real()“. S „file_text_close()“ zatvaramo datoteku. Kod čitanja provjerimo da li datoteka postoji i potom pomoću „file_text_read_real()“ pročitamo podatak i postavimo u objekt „score“. Kod skripte glasi:

```
function scr_save() {
```

```
      if(file_exists("Save01"))
          file_delete("Save01");

      var SaveFile = file_text_open_write("Save01");
      var SaveScore = obj_score.time1;

      file_text_write_real(SaveFile,SaveScore);
      file_text_close(SaveFile);
}
function scr_load() {
      if(file_exists("Save01")){

          var LoadFile = file_text_open_read("Save01");
          var LoadScore = file_text_read_real(LoadFile);

          obj_score.time1 =LoadScore;
          file_text_close(LoadFile);
      }
}
```

5.9. Izbornici

Unutar igre ima više različitih izbornika. Mnogi od njih imaju različite uloge i različite interakcije, ali svi imaju istu generalnu strukturu. Od glavnog izbornika i opcija do izbornika izgleda lika, svima je struktura koda veoma slična. Zbog toga ćemo opisati izbornike na temelju glavnog izbornika.

Svi izbornici imaju objekt koji ih predstavlja. Glavni izbornik ima objekt koji se zove „obj_menu“. Unutar tog objekata imamo nekoliko događaja, ali za sam izbornik su bitna samo dva, a to su događaj kreiranja i događaj koraka. Kod događaja kreiranja glasi:

```
menu[0] = "Arcade mode";
menu[1] = "Versus mode";
menu[2] = "Leaderboard";
menu[3] = "Options";
menu[4] = "Store";
menu[5] = "Credits"
menu[6] = "Quit"
space = 50;
mpos = 0;
```

U događaju kreiranja stvorimo indeksirano polje koje služi kao izbori unutar izbornika. Zatim imamo varijablu „mpos“ koja je skraćena od „menu_position“ i predstavlja trenutni odabir izbornika. Izbornikom se upravlja s tipkama i uvijek je jedna opcija odabrana. Varijabla „space“

sadrži u sebi razmak između izbora izbornika u pikselima. Oni se koriste za crtanje izbornika. Kontrola samog izbornika se izvršava unutar događaja koraka. Njegov kod glasi:

```
scr_input();
var menu_move = 0;
menu_move -= max(menu_up_key,0);
menu_move += max(menu_down_key,0);

if(menu_move != 0){                                    //move up and down
    audio_play_sound(snd_menu_move,11,0);
      mpos += menu_move;
    if(mpos < 0) mpos = array_length_1d(menu) - 1;
    if(mpos > array_length_1d(menu) - 1) mpos = 0;
}
var push;
push = max(action_key,start_key,0);
if(push == 1){                                        //select option
      switch(mpos){
        case 0: audio_play_sound(snd_select,11,0);
            room_goto(rm_arcade);
                  break;
        case 1: audio_play_sound(snd_transition,11,0);
            room_goto(rm_local_mp_start);
                  break;
        case 2: audio_play_sound(snd_transition,11,0);
            room_goto(rm_leaderboard);
                  break;
        case 3: audio_play_sound(snd_transition,11,0);
            room_goto(rm_options);
                  break;
        case 4: audio_play_sound(snd_transition,11,0);
            room_goto(rm_store);
                  break;
        case 6:game_end();
                   break;
    }
}
```

Na početku koda događaja koraka izvodimo skriptu unosa. To je ista skripta koju smo opisali kod kontrole igrača. Skripta funkcionira u ovome slučaju na isti način, ali samo gledamo varijable koje su vezane za izbornik. Za razliku od igrača, kod izbornika držanje tipke daje samo jedan signal umjesto konstantnih pozivanja signala. Varijabla „menu_move“ pomoću „max()“ funkcije provjerava da li imamo pomak ili nemamo. Ako je pomak prema gore, varijabla je negativa, a u slučaju da je prema dolje je pozitiva. Zatim ako varijabla ima vrijednost pustimo zvučni efekt pomicanja izbornika. Mijenja se vrijednost pozicije izbornika za jedan prema više ili prema manje. U istoj petlji provjerimo da li smo premašili raspon polja i postavimo maksimalnu ili minimalnu vrijednost na način da izgleda kao da se izbornikom može kružiti. Varijabla „push“ se koristi za tipke potvrde. Isto kao i prijašnja varijabla koristi „max()“ za prepoznavanje odabira. Zatim unutar grananja, koje provjerava da li je odabir selektiran, imamo switch grananje koje izvodi funkcionalnost ovisno o poziciji izbornika. U velikoj većini odabira prvo pusti zvučni efekt odabira i zatim promijeni sobu. U slučaju odabira izlaska iz igre, pokreće se funkcija „game_end()“ koja završava igru.

Svi objekti izbornika imaju događaj crtanja koji crta polje navedenu u događaju kreiranja. Za crtanje koristi for petlju i razmak definiran među početnim varijablama. Kako bi se mogli snalaziti unutar izbornika, kreirana je mala ikona glave astronauta. Ona služi kao pokazivač na trenutačno odabranu opciju izbornika.

5.10. Kontroliranje glazbe

Svi zvukovi su grupirani u dvije kategorije. Te kategorije su zvučni efekti i glazba. Tu podjelu smo napravili kako bi smo mogli kontrolirati glasnoću jedne grupe bez da utječemo na drugu. To inače nije potrebno raditi jer GameMaker automatski sve zvukove stavlja u jednu unaprijed definiranu grupu. Ta grupa se automatski učitava pri pokretanju igre, no pošto smo mi kreirali posebne grupe, one nisu učitane. Zbog toga ih moramo prije korištenja učitati. Za učitavanje i kontroliranje glazbe smo kreirali poseban objekt za glazbu. Taj objekt smo nazvali „obj_music“ i on je stalan objekt koji je prisutan u svim sobama.

Objekt za glazbu sadrži četiri događaja. Ti događaji su događaj kreacije, alarm, događaj pokretanja sobe i događaj asinkronog učitavanja. Događaj kreiranja pokreče učitavanje grupa zvukova funkcijom „audio_group_load()“ i postavlja varijable učitavanja u ne istinito stanje. Alarm služi kao znak da postavi varijablu učitavanja u istinito stanje nakon što je asinkroni signal učitavanja pristigao. Kod asinkronog signala glasi:

```
if (audio_group_is_loaded(snd_music)){
      audio_play_sound(snd_menu,100,true);
      audio_group_set_gain(snd_music,music_volume,0);
      alarm[0]= room_speed*5;
}
if(audio_group_is_loaded(snd_effects)){
      audio_group_set_gain(snd_effects,effect_volume,0);
      effect_load=true;
}
```

Unutar asinkronog događaja provjeravamo da li je određena grupa zvukova učitana, potom postavimo volumen grupe s funkcijom „audio_group_set_gain()“. Ovisno o grupi, ili pustimo početnu pjesmu i postavimo alarm, ili postavimo da su efekti učitani.

U događaju pokretanja sobe provjeravamo u kojoj smo sobi pomoću „room“ varijable. Kada ustanovimo koje grananje je ispravno, s „audio_stop_all()“ zastavimo trenutno reproduciranu glazbu i nakon toga pustimo one zvukove koje je potrebno pustiti u toj sobi. Primjer jednog grananja je:

```
if(room == rm_arcade){
      audio_stop_all();
      audio_play_sound(snd_game_music1,100,true);
}
```

6. Zaključak

Kreiranje računalne igre je kompliciran i dugotrajan proces. Taj proces ne zahtjeva samo znanje programiranja, već također zahtijeva dizajniranje, kreiranje umjetničkih resursa, narativno pričanje i mnogi drugi. U ovome radu sam najbolje što mogu pokazao svaki od tih znanja.

Iako drugi engine-i daju veću snagu i veće mogućnosti. Pokazao sam da je GameMaker Studio jednako dobar engine. Može proizvesti igre koje su dostojne mnogim nagradama. GameMaker je također savršen za kreiranje prve računalne igre. To mogu iz iskustva reći jer je ovo moja prva računalna igra i s lakoćom sam je kreirao.

Današnje doba dominiraju trodimenzionalne igre, ali i u ovom dobu je moguće kreirati uspješne dvodimenzionalne igre. Čak i one koje ne zahtijevaju veliku i kompliciranu narativnu priču. Sve u svemu kreiranje video igre nije toliko komplicirano kao što neki misle. Samo je potrebna dobra ideja i dovoljno volje.

Popis literature

[1] R. Potr, The growth of the gaming industry in the context of creative industries, preuzeto 25.6.2022.,https://cejsh.icm.edu.pl/cejsh/element/bwmeta1.element.desklight-41940a4b-8b1b-4cb2-9289-12d8b8210b9c

[2]J. Gregory, Game Engine Architecture, Third Edition, preuzeto 25.6.2022., https://www.taylorfrancis.com/books/mono/10.1201/9781315267845/game-engine-architecture-third-edition-jason-gregory

[3] Apache HTTP Server Project, preuzeto 26.6.2022., https://httpd.apache.org/

[4] S. Suehring, MySQL Bible, preuzeto 26.6. 2022., http://box.cs.istu.ru/public/docs/other/_New/Books/Data/DB/MySQL/MySQL%20Bible.pdf

[5] D. Vinciguerra, A. Howell, The GameMaker Standard, preuzeto 27.6.2022., https://books.google.hr/books?id=dM-9CgAAQBAJ&pg=PA2&lpg=PA3&focus=viewport&hl=hr#v=onepage&q&f=false

[6] About YoYo games, preuzeto 27.6.2022., https://gamemaker.io/en/about

[7] R Mokua, W Mwangi, J M Kanyaru, DESIGN AND IMPLEMENTATION OF AN OBJECT ORIENTED PROGRAMING LANGUAGE, preuzeto 27.6.2022., http://journals.jkuat.ac.ke/index.php/jscp/article/view/720

[8] J. Artz, A guide to the SQL standard, preuzeto 27.6.2022., https://dl.acm.org/doi/abs/10.5555/64092

[9] R. Lerdorf, Programming PHP, preuzeto 27.6.2022., https://books.google.hr/books?hl=hr&lr=&id=7OjvOmol3CcC&oi=fnd&pg=PR9&dq=PHP&ots=1sSn3Vc7v0&sig=GqhG9RdPNxzRs5RLWGbF5MM3dTk&redir_esc=y#v=onepage&q=PHP&f=false

[10] GameMaker Maual – Language, preuzeto 28.6.2022., https://manual.yoyogames.com/#t=GameMaker_Language.htm

[11] GameMaker Maual – Local Variables, preuzeto 28.6.2022., https://manual.yoyogames.com/GameMaker_Language/GML_Overview/Variables/Local_Variables.htm

[12] GameMaker Maual – Instance Variables, preuzeto 28.6.2022., https://manual.yoyogames.com/GameMaker_Language/GML_Overview/Variables/Instance_Variables.htm

[13] M. Wolf, The Medium of the Video Game, preuzeto 28.6.2022., https://books.google.hr/books?hl=hr&lr=&id=lKZriBxbcwQC&oi=fnd&pg=PA11&dq=video+game&ots=DIsiIXHFna&sig=D0luTvaONrxH_YiwO6yqkZEeE3I&redir_esc=y#v=onepage&q=video%20game&f=false

[14] C. Kelly, Programming 2D Games, preuzeto 28.6.2022., https://books.google.hr/books?hl=hr&lr=&id=PoPw--vPXfUC&oi=fnd&pg=PP1&dq=2d+games&ots=9wja1Pl364&sig=HlJ75hifBrHtgenYo-plRq9Yaf0&redir_esc=y#v=onepage&q=2d%20games&f=false

[15] Merriam-Webster Dictionary, preuzeto 29.6.2022., https://www.merriam-webster.com/dictionary/arcade%20game

[16] B. Nicoll, Bridging the Gap: The Neo Geo, the Media Imaginary, and the Domestication of Arcade Games, preuzeto 29.6.2022., https://journals.sagepub.com/doi/full/10.1177/1555412015590048

[17] S. Sephr, The Role of Competitiveness in the Cognitive Absorption of Video Games, preuzeto 29.6.2022., https://core.ac.uk/download/pdf/301352626.pdf

[18] K. Chen, Deep Reinforcement Learning of Flappy Bird, preuzeto 29.6.2022., https://cs229.stanford.edu/proj2015/362_report.pdf

[19] Undertale IMDb, preuzeto 29.6.2022., https://m.imdb.com/title/tt5238848/awards/?ref_=tt_awd

Popis slika

Prilozi

Instalacija igre je dostupna na: https://haha-yes.ddns.net/index.php/s/sYNoLAF8WTxRGbP