Izrada računalne igre u okruženju GameMaker

Matić, Vlatko

Sveučilište Jurja Dobrile u Puli
Fakultet informatike u Puli

VLATKO MATIĆ

IZRADA RAČUNALNE IGRE U OKRUŽENJU GAMEMAKER

Završni rad

Pula, rujan 2018.

Sveučilište Jurja Dobrile u Puli
Fakultet informatike u Puli

VLATKO MATIĆ

IZRADA RAČUNALNE IGRE U OKRUŽENJU GAMEMAKER

Završni rad

JMBAG: 0303054126, redoviti student
Studijski smjer: Sveučilišni preddiplomski studij Informatike

Predmet: Napredne tehnike programiranja
Znanstveno područje: Društvene znanosti
Znanstveno polje: Informacijske i komunikacijske znanosti
Znanstvena grana: Informacijski sustavi i informatologija
Mentor: doc. dr. sc. Tihomir Orehovački

Pula, rujan 2018.

IZJAVA O AKADEMSKOJ ČESTITOSTI

Ja, dolje potpisani _____, kandidat za prvostupnika _____ovime izjavljujem da je ovaj Završni rad rezultat isključivo mojeg vlastitog rada, da se temelji na mojim istraživanjima te da se oslanja na objavljenu literaturu kao što to pokazuju korištene bilješke i bibliografija. Izjavljujem da niti jedan dio Završnog rada nije napisan na nedozvoljen način, odnosno da je prepisan iz kojega necitiranog rada te da ikoji dio rada krši bilo čija autorska prava. Izjavljujem, također, da nijedan dio rada nije iskorišten za ikoji drugi rad pri bilo kojoj drugoj visokoškolskoj, znanstvenoj ili radnoj ustanovi.

Student

_____

U Puli, _____, _____ godine

IZJAVA
o korištenju autorskog djela

Ja, _____, dajem odobrenje Sveučilištu Jurja Dobrile u Puli, kao nositelju prava iskorištavanja, da moj završni rad pod nazivom

_____

_____

_____ koristi na način da gore navedeno autorsko djelo, kao cjeloviti tekst, trajno objavi u javnoj internetskoj bazi Sveučilišne knjižnice Sveučilišta Jurja Dobrile u Puli te kopira u javnu internetsku bazu završnih radova Nacionalne i sveučilišne knjižnice (stavljanje na raspolaganje javnosti), sve u skladu s Zakonom o autorskom pravu i drugim srodnim pravima i dobrom akademskom praksom, a radi promicanja otvorenoga, slobodnoga pristupa znanstvenim informacijama.
Za korištenje autorskog djela na gore navedeni način ne potražujem naknadu.

U Puli, _____ (datum)

Potpis

_____

Sadržaj

1.Uvod

Računalne igre su vrsta video igara za osobna računala. Mogu biti višestruko složenije od igara za pametne telefone i tablete, dok su sličnije igrama za konzole. Velike igre su često i „cross platform“, što znači da se radi o istim naslovima za konzole i za osobna računala. Igre za osobna računala lakše je modirati, te su kao takve popularnije među igračima i programerima koji su ljubitelji modiranja igara.

Prednost računalnih igara je i u tome što si igrači sa snažnijim računalima mogu prilagoditi „jače“ postavke, poput bolje grafike. S druge strane, jedan od glavnih nedostataka računalnih igara je taj što su osobna računala namijenjena za puno veći spektar različitih radnji od konzola, te je ponekad igre potrebno izmijeniti kako bi ih se moglo prilagoditi osobnom računalu. Konzolama je prioritet biti kvalitetno konfiguriran s video igrama, s obzirom na to da im je to glavna svrha.

Prema procjeni „Newzoo“, magazina specijaliziranog za tržište video igara, računalne igre su 2016. godine predstavljale nešto manje od trećine vrijednosti ukupnog tržišta video igara. Udio računalnih igara na svjetskom tržištu smanjuje se iz godine u godinu, jer tim ukupnim tržištem sve više prevladavaju igre za pametne telefone. Procjena za 2018. godinu je svega 26%. Usprkos tome, vrijednost tržišta računalnih igara je u rastu, samo što je taj rast sporiji od rasta tržišta igara za konzole i pametne telefone (Wijman, 2017.).

Kompanije koje izdaju igre uglavnom ih rade u dogovoru s vanjskim suradnicima (eng. outsourcing) ili vlastitim kompanijama specijaliziranima za razvoj igara. Vanjski suradnici uglavnom su manje kompanije za razvoj igara. Najveće i najpoznatije izdavačke kompanije video igara, su veoma poznati: EA, Bethesta, Ubisoft, Rockstar, Activison, Blizzard, itd.

Zadatak ovog rada bio je izraditi igru u radnom okruženju Game Maker Studio. Igra izrađena za ovaj projekt nazvana je „Alpha-Bravo“. Naziv „Alpha-Bravo“ simbolizira cilj igre, a to je prelazak s početne točke „A“ na krajnju točku „B“, u žargonu vojne abecede. Igru se može svrstati u žanr pucačkih i platformer igara.

Pokretanjem igre otvara se početni izbornik u kojem igrač može birati različite mogućnosti: pokretanje nove igre, učitavanje već započete igre, otvaranje prozora s opcijama (odabir tipki za igru i kontrola zvuka) i izlaz iz igre.

Igrač ima ulogu vojnika na zadatku u pustinji. Prolazeći pustinjom, igrač nailazi na neprijateljske vojnike koji ga pokušavaju ubiti. Cilj igre je prolazak kroz pustinju u koju je igrač iskrcan na početku. Na kraju pustinje je obala. Pritom je potrebno obraniti se od neprijatelja, izbjeći prepreke i signalizirati svoju lokaciju na kraju svake od tri razine. Neprijateljski vojnici napadaju igrača kao pješaci ili u vozilu, a prepreke predstavljaju nagazne mine i razne zamke u podu. Igra se odvija u pustinji i u raznim neprijateljskim utvrdama.

1.1. Analiza postojećih srodnih igara

Glavni izvori inspiracije projekta „Alpha-Bravo“ bile su igre Nintedov „Super Mario Bross“ iz 1985. i Konamijeva „Contra“ iz 1987. Obje igre slične su ovom projektu prvenstveno zbog toga što su 2D side-scroller-i retro dizajna. Retro izgled tih igara je dokaz da igra može biti vrlo popularna iako ne izgleda moderno. Takve igre bude nostalgiju u igraču i ističu se među modernim igrama današnjice.

„Super Mario Bross“ iz 1985. prva je igra franšize „Super Mario“, a druga franšize „Mario“ koja je najprodavanija franšiza u povijesti. Procjenjuje se da je franšiza „Mario“ prodana u preko pola milijarde primjeraka. „Super Mario Bross“ iz 1985. prodan je u otprilike 40 milijuna primjeraka, što ga čini jednom od najprodavanijih igara u povijesti. (Lisa, 2017.)

Igrač u ulozi Super Maria bori se s različitim neprijateljima u carstvu gljiva kako bi prešao razinu i stigao do jednog od dvoraca. Cilj igre je doći do dvorca u kojem se nalazi princeza koju treba spasiti. Neprijatelje pobjeđuje tako što skoči na njih ili ispali neki od privremenih projektila koje skupi za vrijeme igre.

Dijelovi projekta koji su inspirirani ovom igricom su: sustav bodovanja koji nema veliku značajnost, ali ipak budi u igraču želju za što boljim rezultatom. Jedna od ikoničnih scena Super Maria je scena zamjene neprijateljske zastave kad Mario priđe dvorcu. Mario spusti neprijateljsku zastavu s koplja, a na dvorac podigne svoju (slika: 1). U ovom projektu zastave se nalaze na raznim lokacijama te zamijena zastava služi za dobivanje dodatnih bodova. Odvijanje radnje, to jest tempo igre „Alpha-

Bravo“ sličan je tempu Super Maria, za razliku od „Contre“ koja se odvija puno brže, s velikom količinom neprijatelja i njihovih projektila. Super Mario Bross za razliku od Contre i Alpha-Bravo sadrži vremenski brojač koji ograničava vrijeme igre.

Slika 1: Super Mario Bross (1985.)

„Contra“ je izdana 1987. godine i u to vrijeme bila je jedna od najpopularnijih igara na svijetu. U svoje vrijeme bila je posebno prepoznatljiva po tome što su je mogla igrati dva igrača odjednom. Jedan igrač bio bi u ulozi Billa, a drugi igrač bi bio Lance. (The International Arcade Museum, 2017.)

Radnja igre odvija se u budućnosti na otoku gdje se nalazi zla korporacija s namjerom da osvoji svijet.

Contra, iako nije bila toliko revolucionarna kao Super Mario, zapamćena je kao jedna od najtežih igara iz tog doba. Kako bi igrač prešao neke razine potrebno ih je dobro poznavati što je sličnost ovom projektu gdje igrač mora naučiti prepoznati mine, sigurne zaklone, uočiti neprijatelje... Alpha-Bravo s Controm dijeli mnogo sličnosti kao što su potreba za saginjanjem, trčanjem i izbjegavanjem projektila. Contra ne nudi zaklone već igrača potiče na stalno kretanje kako bi „preživio“. Zajednička

karakteristika Super Maria i Contre je ta što igrač ima više „života“ to jest ima ograničen broj puta koliko može ponavljati istu razinu prije nego bude vraćen na početak. No Alpha-Bravo igraču dopušta neograničen broj pokušaja za prelazak iste razine.

Slika 2: Contra (1987.)

1.2. Usporedba Game Makera s drugim radnim okruženjima za PC-igre

Platforme za izradu igara (eng. game engine) počele su se pojavljivati sredinom 1990-ih paralelno pojavi prvih 3D video igara. One služe kako bi se ubrzao i pojednostavio proces razvoja video igara. Prve video igre bilo je potrebno razvijati „od nule“ što znači da je svaki element igre bilo potrebno kodirati iz početka. Platforme za izradu igara grafički prikaz, zvučni sustav i osnovne logičke elemente koje sama platforma definira razdvajaju od pravila igre i umjetničkih elemenata kao što su naprimjer zvučne ili slikovne datoteke koje unosi, kreira i kodira programer. Dakle, platforme za izradu igara programeru nude temelje na kojima on gradi igru. Platforme se prvenstveno razlikuju po složenosti. Složenije i manje složene razlikuju se po tome što složenije platforme nude manje detaljne temelje i okvire igre te zbog toga nude mogućnosti za kompleksnije igre. Dok jednostavniji sadrže detaljnije predloške s kojim je puno lakše raditi, ali te platforme nude puno manje mogućnosti razvoj igre.

Platforme se razlikuju i po tome što su često specijalizirane za pojedini žanr igara. Platforma dizajnirana za boksačku igru za dva igrača biti će znatno drugačija od one za veliku grupu online igrača ili od ratne igre iz prvog lica ili strategije. S druge strane, takve platforme i dalje mogu imati mnogo zajedničkih karakteristika kao što su sličan sustav za grafički prikaz ili sličan zvučni sustav (Gregory, 2018.).

Game Maker spada među srednje zahtjevne platforme za izradu igara te je kao takav relativno jednostavan za korištenje, ali i relativno ograničen. Prvenstveno je namijenjen izradi 2D igara. Postoji mnogo drugih 2D platformi kao što su „Game Salad“ i „Stencyl“. One su puno jednostavnije za početnike, ali time i ograničenije. Te dvije platforme namijenjene su za početnike kojima omogućuju da nauče osnove izrade igara i programsku logiku.

„Game Salad“ sadrži isključivo „drag and drop“ metodu te ne nudi mogućnost kodiranja. „Game Salad“ sadrži „svoju fiziku“ koju se može izmjenjivati i „pravila“ koja se zatim pridružuje kreiranim objektima (Tanant, 2018.).

„Slika 3“ sadrži prikaz izrade igre u „Game Salad“ okruženju. Na gornjem dijelu lijeve strane slike mogu se vidjeti kreirani objekti koji se dodaju u igri, a ispod njih nalazi se

popis funkcija koje se pridružuju tim objektima. Na desnoj strani slike prikazan je trenutni izgled igre.

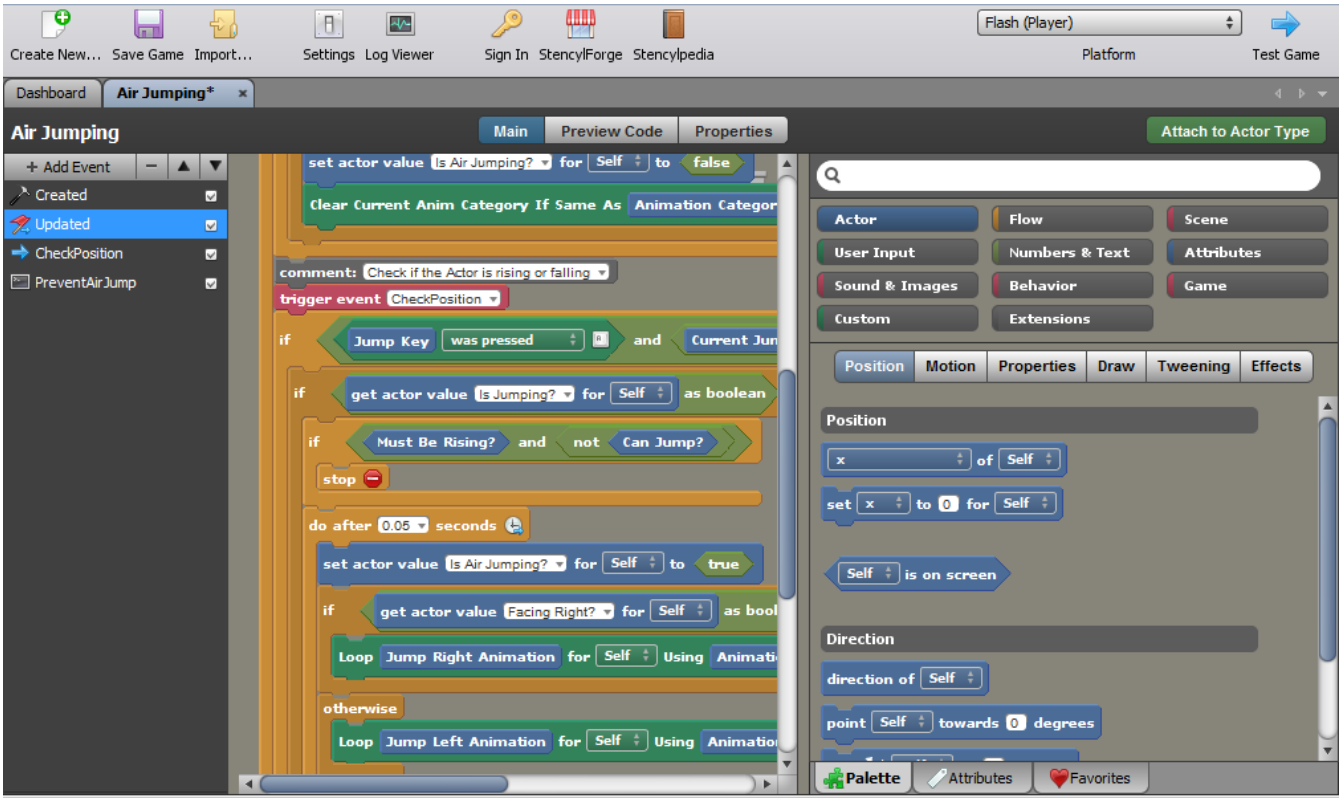
„Game Salad“ od „Game Maker“-a razlikuje se po tome što ne nudi mogućnost kodiranja, već je više namijenjen početnicima bez predznanja. U „Game Maker“-u programer sam mora definirati „fiziku“ koristeći kôd ili predloške. „Game Salad“, kao što je već rečeno, sadrži ugrađenu fiziku koju se može tek izmjenjivati što programeru nudi slabije mogućnosti. „Game Salad“ mnogo je jednostavniji od „Game Maker“-a i kao takvog lakše ga je naučiti korisititi, ali zato nudi manje mogućnosti za budući razvoj znanja.

Slika 3: Game Salad

„Stencyl“ je baziran na vizualnom programskom jeziku „scratch“ koji je razvio tim sa Sveučilišta MIT. „Scratch“ sadrži gotove linije kôda koje se slažu poput slagalice. Iako se koristi metoda „drag and drop“, „Stencyl“ je odlična platforma za učenje programiranja jer prikazuje kôd te tako pomaže razumijevanju kôda (Tanant, 2018.).

„Slika 4“ prikazuje kodiranje jezikom „scrath“. Na lijevoj strani slike nalazi se lista „događaja“ i ukoliko do njih dođe, aktivira se priloženi kôd koji je vidljiv na sredini slike. Na gornjem dijelu desne strane slike nalazi se popis različitih vrsta funkcija. Nakon što je odabrana željena vrsta funkcija otvara se prozor s funkcijama odabrane vrste koje korisnik zatim ubacuje u glavni kôd.

Najistaknutija karakteristika „Stencyl“-a je ta što je primarno stvoren za edukaciju početnika, dok „Game Maker“ nudi mogućnosti i za relativno iskusnije programere.



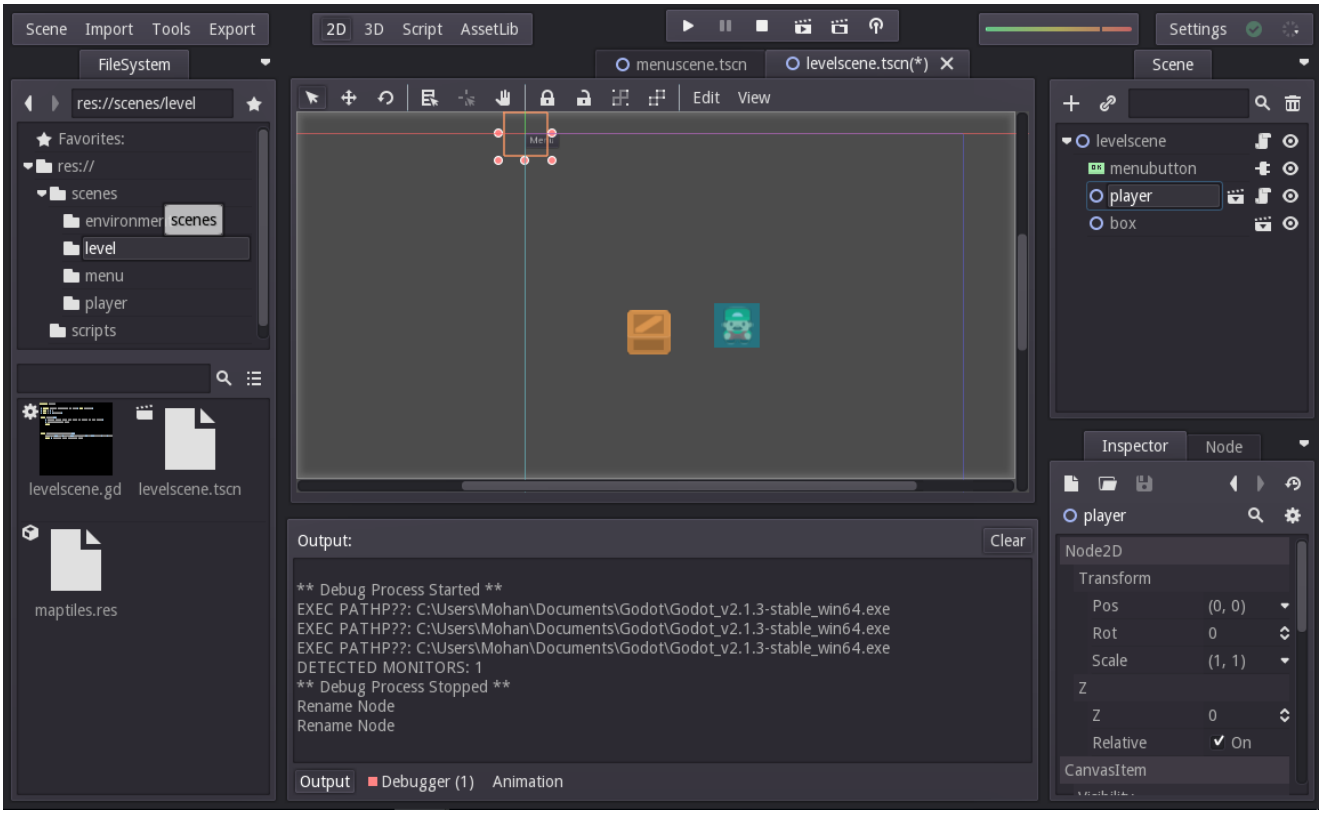
Slika 4: Stencyl

„Godot“ je jedna od najpopularnijih „open source“ platformi za izradu igara. Za razliku od prethodnih sučelja nudi izradu igara u 2D-u i u 3D-u. „Godot“ ne sadrži „drag and drop“ metodu izrade igre, već je potrebno razumjeti programiranje i to zahtjeva određenu količinu predznanja.

„Slika 5“ prikazuje izradu dvodimenzionalne igre u „Godot“-u. Na sredini slike prikaz je odabrane scene u koju programer postavlja objekte. Na gornjem dijelu lijeve strane slike nalazi se popis raznih scena koje se nalaze u projektu. Ispod toga nalazi se popis vanjskih datoteka korištenih u projektu. Na desnoj strani slike nalazi se lista čvorova u trenutnoj sceni, a ispod toga prozor s postavkama odabranog čvora.

Čvorovi u „Godot“-u predstavljaju klasu za sve objekte u sceni. Čvorove se može slagati hijerarhijski u odnos djece i roditelja što rezultira „strukturom stabla“. (Docs Godot, 2018.).

„Godot“ se razlikuje od „Game Maker“-a po tome što zahtjeva određenu količinu predznanja, prvenstveno zbog toga što ne nudi mogućnost „drag and drop“ metode, već se isključivo oslanja na programski kôd.



Slika 5: Godot

„Game Maker“ se pokazao kao najbolja platofrma za ovaj projekt zbog toga što pruža ravnotežu jednostavnosti za početnike i slobode za stvaranje novih igrica za iskusne. Zbog mogućnosti kodiranja „Game Maker“ omogućava izradu kompliciranijih igara od „Game Salad“-a i „Stencyl“-a, a mogućnost korištenja „drag and drop“ metode početniku olakšava snalaženje u izradi igara.

2. Game Maker

„Game Maker studio“ je platforma za izradu jednostavnijih igara koja korisnicima omogućava jednostavnu izradu igara „drag and drop“ metodama i jednostavnim sučeljem (Vinciguerra, 2018.).

Game Maker razvijen je od strane „Yoyo games“. Prva verzija Game Maker-a razvijena je 1999. godine. Igre u Game Maker-u mogu se izrađivati za različite platforme: Microsoft Windows, Play Station4, Ubuntu, MacOs, Android, iOS, XboxOne... Glavni fokus Game Maker-a je izrada 2D igara, no može ih se izrađivati i u 3D-u. Game Maker se koristi vizualnim programiranjem (drop and drag) i skriptnim programiranjem (GML). Zbog jednostavnog sučelja mogu ga koristiti hobisti i početnici, a zbog velikog spektra izbora, i profesionalci.

2.1. Rad u Game Maker-u

Nakon prvog pokretanja Game Maker-a, odabire se mogućnost „New Project“. Zatim je potrebno odabrati ime novog projekta te odabrati lokaciju na računalu gdje će projekt biti spremljen. Klikom na tipku „Create“ otvara se novi prazni projekt.

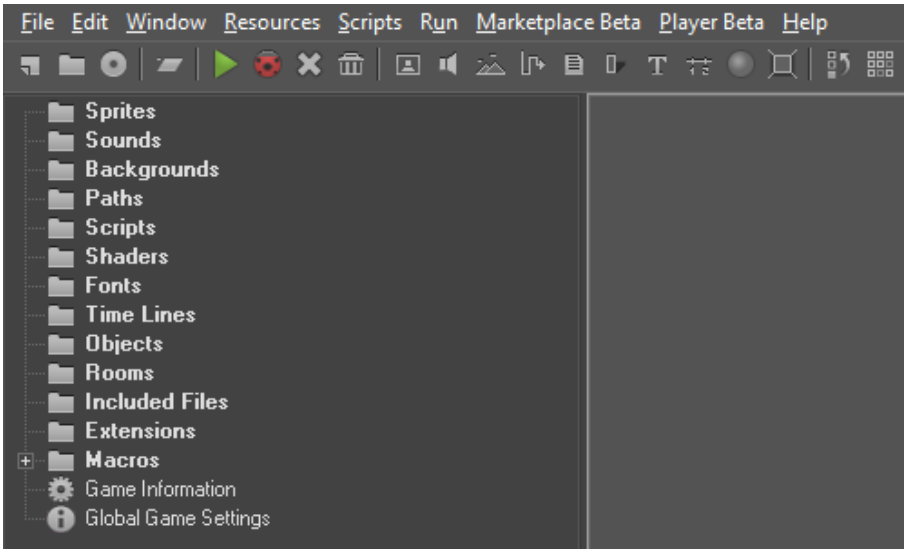
Alatna traka na vrhu prozora nudi mnoge alate (slika: 6). Važniji alati su: file, edit, resources, scripts i run.

„File“ služi za upravljanje projektom, na primjer otvaranja novog projekta, učitavanje postojećeg projekta, spremanje promjena trenutnog projekta, itd.

„Edit“ označava uređivanje datoteke, kao što je brisanje ili udvostručavanje označenog teksta, pretraživanje po riječima, otvaranje prozora sa svojstvima odabranog objekta, itd.

„Resources“ je prečac za kreiranje novih resursa, poput sprite-ova, zvukova, pozadina, skripti, objekata, itd. Isti prečaci nalaze se i u četvrtom odjeljku druge alatne trake.

„Scripts“ nudi prikaz različitih programskih varijabli i funkcija...



Slika 6: Alati

Sljedeća alatna traka nudi alate koji su ponuđeni i na prvoj alatnoj traci, ali tu su probrani oni koji se češće koriste. Stvaranje novog projekta, spremanje promjena, pokretanje igre, „debug mode“, kreiranje različitih resursa, itd.

S lijeve strane nalaze se mape nazvane po vrstama resursa koji se u njih spremaju. Ako u mapi ima spremljenih resursa, ona se „otvori“ lijevim klikom te se izlistaju spremljeni resursi. Desnim klikom otvara se izbornik koji nudi kreiranje novog resursa tipa odabrane mape ili kreiranje nove mape unutar postojeće. Klikom na neki od resursa, s desne strane otvara se novi prozor tog pojedinog resursa. Prozori resursa sadrže opis resursa, njegove karakteristike i ostale mogućnosti koji se izmjenjuje i nadodaje.

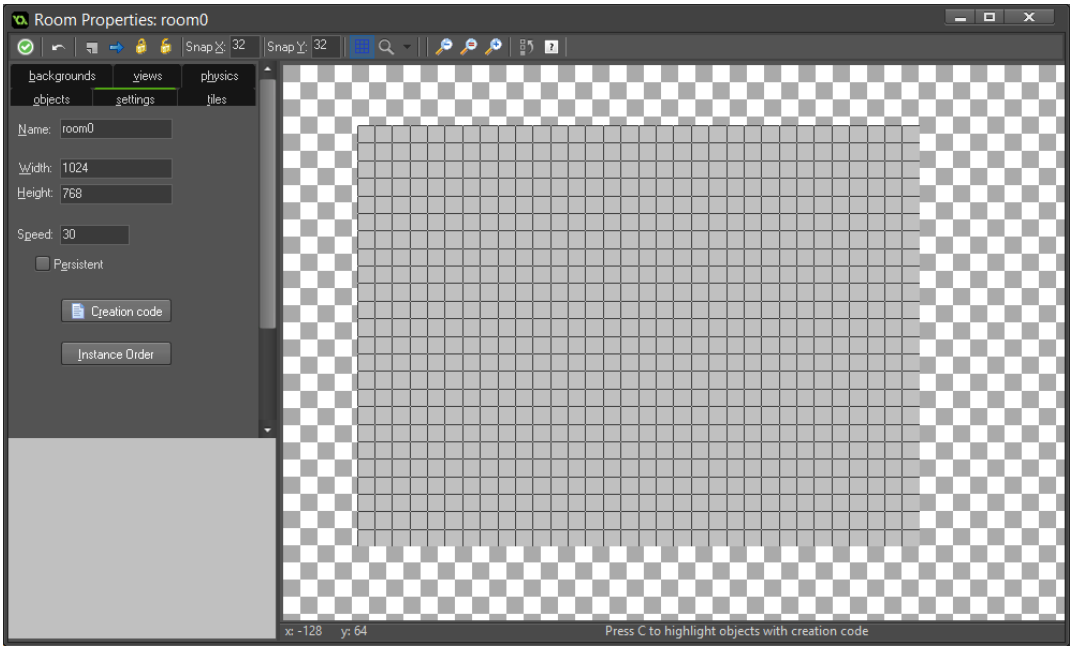
2.2. Resursi

Resursi u Game Maker-u označavaju različite vrste datoteka unesenih u program i kreiranih u programu. To su zvukovi, slike, pozadinske slike, objekti, skripte, fontovi, sobe...

2.2.1. Soba (room)

Prvi korak nakon otvaranja novog projekta je kreirati barem jednu sobu (eng. room), kako bi se igra mogla pokrenuti. Sobe u Game Makeru predstavljaju grafičko sučelje igre, to jest prostor u kojem se sve događa. Sobe se sastoje od dvodimenzionalnog prostora reguliranog različitim parametrima. Nove sobe kreiraju se prazne (slika: 7).

Parametri sobe prikazani su u izborniku na lijevoj strani prozora sobe u alatu „settings“. Parametri su: veličina sobe (visina i širina) izražena u pikselima, naziv sobe i brzina. Brzina (eng. speed) sobe označava učestalost izvršavanja koda u jednoj sekundi: ukoliko je brzina sobe postavljena na 30, cijeli kod, koji se nalazi u sobi, izvršit će se 30 puta u jednoj sekundi. Gornji lijevi kut sobe je početna točka (x=0, y=0). Vrijednost „x“ koordinate raste u desnu stranu, a „y“ koordinate prema dolje (docs.yoyogames, 2018.).



Slika 7: prazna soba

Soba može imati i „creation code“ koji se izvršava u trenutku generiranja sobe, a prije izvršavanja kôda objekata. Osim parametara, u izborniku se nalaze i drugi alati pomoću kojih se mogu izmjenjivati i objekti, pozadine, pločice (eng. tiles), fizika i pogledi (eng. views).

Objekti se u sobu postavljaju klikom na željenu poziciju u prostoru. Odabiru se u alatu „objects“ u kojem se pojedinačnim objektima mogu još dodatno izmjenjivati širina i visina, boja, kut zakrivljenosti i prozirnost.

Alat „backgrounds“ (2.2.4.) služi za dodavanje pozadinskih slika u sobu. Pozadinskim slikama može se dodijeliti brzina (zadana vrijednost brzine je 0), promijeniti dimenzije i početna točka. Ukoliko se u sobu postavi više pozadina, potrebno je odrediti hijerarhiju pozadina, to jest koja treba ići iznad koje.

„Views“ je alat koji služi za upravljanje „pogledima“. Pogled je mehanizam koji omogućuje prikaz odabranog djela ekrana. Pogled se u dvodimenzionalnim igrama koristi kad je objekt igrača premalen da bi bio vidljiv ukoliko se prikazuje cijela razina. U većini slučajeva, pogledu je zadano da prati objekt kojim upravlja igrač. U „views“ alatu može se definirati više pogleda te im se mogu izmjenjivati dimenzije, rezolucija, objekt koji prate, brzina praćenja tog objekta te pozicija u kojoj praćeni objekt treba biti.

2.2.2. Sprite

Sprite-ovi su resursi sa slikom ili animacijom (skupom slika) koji se pridružuju objektu. Prvenstveno služe kao vizualni prikazi objekata. Kako bi bio vidljiv, objektu je potrebno pridružiti sprite.

Nakon što se napravi sprite, potrebno mu je pridružiti jednu ili više slika. Ukoliko se sprite-u pridruži više slika, svaka pojedinačna slika postaje pojedini „frame“ animacije. U Game Maker se mogu uvesti (eng. import) slike mnogih vrsta. Osim klasičnih (.png, .jpeg) i sličnih datoteka, mogu se unositi vektorske slike i skeletalne animacije. Game Maker sadrži „Image Editor“, potprogram za uređivanje ili crtanje slika i animacija.

Sprite-u je potrebno odrediti glavnu točku, odnosno početnu koordinatu. Kada se sprite pridruži objektu, glavna točka predstavlja koordinate na kojima se objekt nalazi. Sprite-u se pridružuje „maska“ za kolizije. Maska, bez obzira na sliku sprite-a kojem je pridružena, predstavlja skup točaka koje igra smatra tim sprite-om. To znači da može doći do kolizije objekata i kad to igrač to ne vidi. Maska se može prilagoditi kako bi imala oblik jednak obliku slike sprite-a ili drugačiji oblik. Zadani oblik je pravokutnik širine i visine jednakih najdaljim točkama od središnje točke sprite-a.

2.2.3. Zvukovi (sounds)

Nakon kreiranja resursa zvuka pridružuje mu se datoteka s računala i odabire ime. Zvukovi služe kao zvučni efekti ili kao glazbena pozadina u igri. Zvukove uvedene (eng. imported) u Game Maker ne pridružuje se objektima, već ih se poziva kroz funkcije (xxx). Uvedenom zvuku može se izmjenjivati frekvencija, brzina, kvaliteta te ga se može prilagoditi za 3D prostor ili kao mono zvuk.

2.2.4. Pozadine (backgrounds)

Pozadine, kao i sprite-ovi, temelje se na slikama. Kreiranoj pozadini može se izmjenjivati naziv i slika. Može se odabrati hoće li pozadina biti korištena kao velika slika ispred koje se generiraju objekti ili kao set „pločica“ (eng. tiles), malih pozadina koje se međusobno nadovezuju kao set pločica. Pozadinu se odabire za svaku pojedinu sobu. U alatu „backgrounds“, pozadini se mogu dodjeljivati neke karakteristike koje ne utječu na samu datoteku pozadine.

2.2.5. Fontovi (fonts)

Resurse fonta ne može se unositi u Game Maker sa računala, već se isključivo kreiraju i izmjenjuju u samom Game Maker-u. Kreirani font može se pridružiti string-u koji se ispisuje u igri pomoću neke funkcije (2.3.2.). Uloga fonta isključivo je estetska. Fontu se dodjeljuje naziv koji služi za kasnije pozivanje fonta. Karakteristike fonta su vrsta teksta („font“), veličina, kosina, debljina i kvaliteta slova. Boja slova se određuje u funkciji u kojoj se i font poziva, a ne u samom fontu.

2.2.6. Objekti (objects)

Objekti su elementi igre koji se nalaze u sobi i izvršavaju većinu radnji. Kako bi se u igri išta moglo odvijati, potrebni su objekti. Objekti mogu imati pridružen sprite koji omogućava njihovu vizualizaciju unutar sobe za vrijeme igre.

Objekti imaju mnoge karakteristike osim samih zadataka koje izvršavaju. Može ih se postaviti da budu nevidljivi bez obzira na to imaju li sprite. Mogu biti „čvrsti“ (eng. solid) i kao takvima može im se pridružiti funkcija da ne mogu prolaziti kroz druge čvrste objekte. Objektima se može zadati karakteristika „upornost“ (eng. persistent) koja pamti promjene u objektu nakon promjene sobe. Ukoliko objekt nije uporan, sve radnje i vrijednosti varijabli restartiraju mu se na početnu vrijednost. Dubina objekta označava koji objekti se crtaju ispred, a koji iza. Zadana vrijednost dubine je 0. Objekt s dubinom 1 crta se iza onoga s dubinom 0, a objekt s dubinom -1 ispred onoga s 0. Objekti mogu nasljeđivati karakteristike drugih objekata. Objekt koji od drugog nasljeđuje karakteristike, „dijete“ je tog objekta, a objekt od kojeg se nasljeđuje je „roditelj“.

Objekti se ne nalaze u igri, već predstavljaju šablon za instance koje se postavlja u sobe (docs.yoyogames, 2018.).

Objekti koji vrše radnje, sadrže programski kôd i događaje (eng. events). Radnje objekata organizirane su prema događajima koje objekt sadrži. Događaji su različite moguće situacije za vrijeme trajanje igre. Ukoliko do njih dođe, aktiviraju kôd koji im je pridružen. Događaji se provjeravaju za svaku instancu objekta koja se nalazi u sobi zasebno, za svaki trenutak (eng. frame) postojanja te sobe. Postoji mnogo različitih vrsta događaja.

Create event. Instanca objekta aktivira „create event“ prvi „frame“ nakon što je kreirana u sobi. Instanca može biti postavljena ručno u sobu te se kao takva kreira zajedno sa sobom, a može biti i naknadno stvorena od strane nekog drugog objekta. Create event je koristan za definiranje varijabli koje će objekt koristiti za svog postojanja i za provjere nekih unaprijed zadanih postavki koje ne moraju biti uvijek jednake u trenutku stvaranja objekta. Izvršava se samo jednom, za svaku pojedinu instancu objekta.

Destroy event. „Destroy event“ se aktivira nakon uništenja instance objekta. Ona može biti uništena pozivom na funkciju za uništenje instanci. Tu funkciju instanca može pozvati za samouništenje ili za uništenje instance drugog objekta, kojoj se onda aktivira „destroy event“ ukoliko ga ima. Za svaku zasebnu instancu, izvršava se samo jednom ukoliko je instanca uništena za vrijeme trajanja sobe.

Step event se aktivira svaki trenutak postojanja instance unutar trenutne sobe. Služi za radnje koje se trebaju izvršavati ili ispitivati cijelo vrijeme dok ta instanca postoji. Izvršavanje „step event“-a ovisi o vremenu postojanja instance koja ga sadrži i o brzini sobe. To je najlakše opisati sljedećom formulom:

(n)= broj aktiviranja „step event“-a.

(t)= vrijeme postojanja instance u sekundama.

(fps)= brzina sobe, broj trenutaka u sobi (eng. frame) u sekundama.

(n)= (t)*(fps)

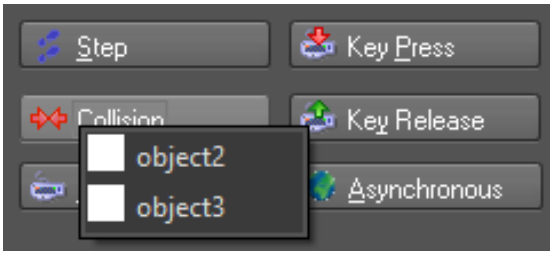
Alarm event. Alarm u GML-u postoji kao varijabla kojoj se dodjeljuje brojčana vrijednost. Nakon definiranja vrijednosti, svaki trenutak sobe (eng. frame) ta se

vrijednost smanjuje za jedan. „Alarm event“ aktivira se kad je vrijednost odgovarajućeg alarma jednaka 0. -1 je vrijednost neaktiviranog alarma. Alarmi se koriste kad je nakon nekog događaja potrebno izvršiti odgođenu radnju. Slika (8) prikazuje primjer aktivacije alarma. Alarm iz primjera nazvan je „alarm[2]“. Taj alarm će za 60 trenutaka aktivirati događaj „Alarm 2“. Ukoliko je brzina sobe jednaka trideset, taj događaj će se aktivirati 2 sekunde nakon aktivacije.

```
1
2 alarm[2]=60;
```

Slika 8: primjer deklariranja alarma

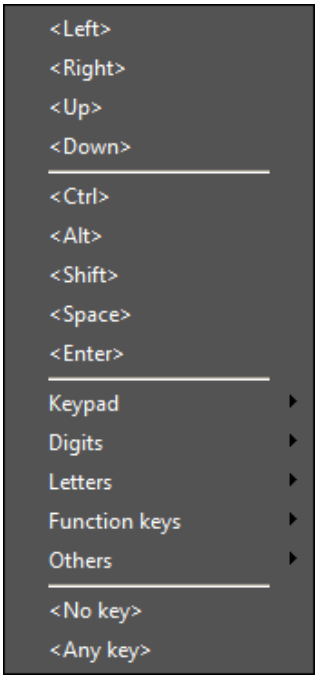
Collision event. „Collision event**“** aktivira se „sudarom“ dvaju objekata. To je trenutak kad se maske dva sprita preklope u dvodimenzionalnom prostoru sobe. Prilikom kreiranja „collision event“-a, potrebno je definirati s kojim drugim objektom se treba „sudariti“ kako bi se aktivirao taj događaj (slika: 9).

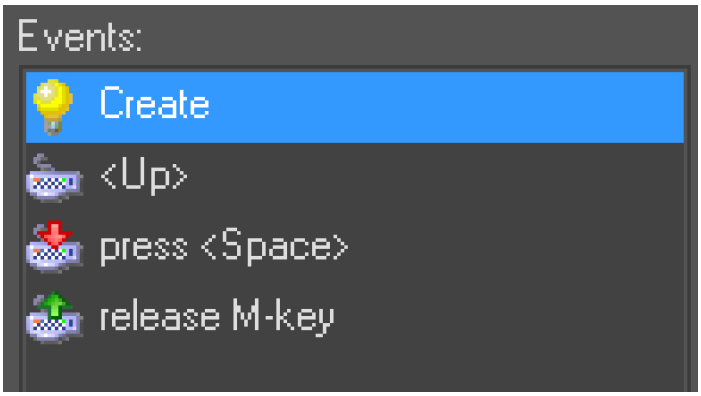


Slika 9: odabir između objekata „object2“ i „object3“

Keyboard events. U prijevodu znači „događaji tipkovnice“. Oni reagiraju na tipkovnicu te kao takvi omogućuju igraču kontrolu nad igrom. Dijele se u tri različite kategorije: **keyboard pressed event**, **keyboard event** i **keyboard released event**. Odabirom jednog od tih događaja otvara se novi prozor (slika: 10), koji sadrži popis svih tipki na tipkovnici. Klikom na jednu od tih mogućnosti kreira se događaj tipkovnice za odabranu tipku. „Keyboard pressed event“ aktivira se u trenutku pritiska na odabranu tipku i traje samo jedan trenutak, to jest izvršava se samo jednom. „Keyboard event“ također se aktivira u trenutku pritiska na tipku, no izvršava se cijelo vrijeme dok igrač odabranu tipku drži pritisnutom. „Keyboard release event“ aktivira se u trenutku kad igrač otpusti odabranu tipku, bez obzira na to kad je pritisnuta, izvršava se jednom. Slika (11) prikazuje tri događaja tipkovnice: „keyboard event“ za

tipku „up“, „keyboard pressed event“ za tipku „space“ i „keyboard release event“ za tipku „M“.





Slika 10: popis tipki

Slika 11: primjer događaja tipkovnice

Draw event. Postoji više podvrsta „događaja crtanja“, no u ovom projektu korišten je samo standardni, „draw event“. Događaj crtanja služi za „ručno“ crtanje. Ručno crtanje je mogućnost crtanja sprite-ova, tekstova, pozadina, geometrijskih tijela i sl. bez obzira na lokaciju objekta unutar sobe.

Other events je kategorija „ostalih događaja“ koji ne pripadaju ni u jednu od prethodnih vrsta događaja. U projektu korištena su četiri događaja iz te kategorije. Svi događaji navedeni u ovoj kategoriji izvršavaju se jednom nakon aktivacije. **„Outside room“** je događaj koji se aktivira kad objekt napusti prostor sobe. Klikom na **„outside view“** otvara se prozor koji prikazuje listu pogleda (eng. view), te je potrebno odabrati željeni pogled. Događaj se aktivira kad objekt napusti pogled. **„Room start“** se aktivira prvi trenutak nakon što se soba otvori, ali izvršava se nakon „create event“-a što znači da može ovisiti o varijablama i drugim stvarima koje se definiraju u „create event“-u (docs.yoyogames, 2018.). **„Room end“** se aktivira zadnji trenutak postojanja sobe.

2.3. GML

GML, je skraćeno od „Game Maker Language“.

Kako bi objekti mogli nešto izvršavati, u događaje im je potrebno dodati radnje. Radnje se mogu dodati „drag and drop“ metodom ili kodiranjem. „Drag and drop“ metoda označava skup radnji gdje se postojeće zadatke za objekt „ubacuje“ u događaj u željenom poretku kako bi se kreirala radnja. No ta metoda je ograničena na mali broj mogućnosti te je zbog toga poželjnije koristiti programski kôd koji ih nudi puno više. Game Maker sadrži svoj posebno kreiran programski jezik, GML. Kako bi se pristupilo prostoru za upis GML kôda, u događaj je potrebno postaviti „execute code“ radnju.

GML je prevedeni programski jezik koji je sintaktički sličan programskim jezicima poput C, C++ i Python-a. S obzirom na to da je interpretirani jezik, GML linije kôda prevode se u izvršivi format svaki puta kada se program izvršava. Zbog toga GML prilikom izvršavanja programa mora koristiti „prevoditelja“ te se izvršava nešto sporije od drugih programa koji se pišu u izvršivom formatu (Ford, 2010.).

2.3.1. Varijable

Prilikom deklariranja varijabli u Game Maker-u nije potrebno deklarirati i tip varijable (integer, string, character, itd.), već je Game Maker sam svrsta u određeni tip prema vrijednosti koja joj je pridružena. U Game Maker-u postoje kategorije varijabli, koje razlikuju varijable prema tome čemu pripadaju.

Kategorije varijabli su sljedeće:

-**instance**: najčešći tip varijabli, definiraju se unutar instance. Te varijable su jedinstvene za svaku instancu i mogu se koristiti za svaki događaj i svaku funkciju te instance. Ukoliko instance varijablu poziva neki drugi objekt kojem ona ne pripada ili je se deklarira u nekom objektu za drugi objekt, potrebno je navesti ime objekta kojem pripada ili treba pripadati i ime varijable.

-**local**: kako bi se deklariranu varijabli svrstalo u local, potrebno joj je prilikom deklaracije ispred imena postaviti funkciju „var“. Local varijabla može se koristiti samo u događaju objekta ili skripti u kojima je deklarirana.

-**global**: varijabla pripada igri, a ne pojedinoj instanci. Svaka instanca može provjeravati ili izmjenjivati varijablu ovoga tipa ravnopravno. Kako bi varijabla bila global varijabla prilikom deklaracije, potrebno joj je ispred imena dodati funkciju „global“.

-**built in variables**: u nastavku rada biti će referencirane kao „programske varijable“. To su varijable koje su automatski ugrađene u objekte i sobe. Postoje isključivo kao varijable kategorija instance ili global. Prepoznaje ih se po tome što se njihovo ime pojavljuje u crvenoj boji (docs.yoyogames, 2018.).

Slika (12) prikazuje načine deklariranja različitih tipova i kategorija varijabli. Varijable „prva“ i „druga“ spadaju u kategoriju instance varijabli. „Prva“ je tipa integer, a druga tipa string. „Treca“ varijabla kategorije je local, a tipa integer. Varijabla „cetvrta_varijabla“ primjer je varijable global kategorije, a tipa je string. Varijable „x“ i „image_yscale“ su programske varijable. Varijabla „sedma“ je varijabla tipa integer, koja ne pripada objektu u kojem se izvršava kôd, već objektu „ob_objekt“, to je primjer pozivanja varijable kategorije instance drugog objekta.

action

```
1 prva=12;
2 druga="vrijednost";
3 local.treca=1221;
4 global.cetvrta_varijabla="druga vrijednost";
5 x=56;
6 image_yscale=30;
7 ob_objekt.sedma=56;
8
```

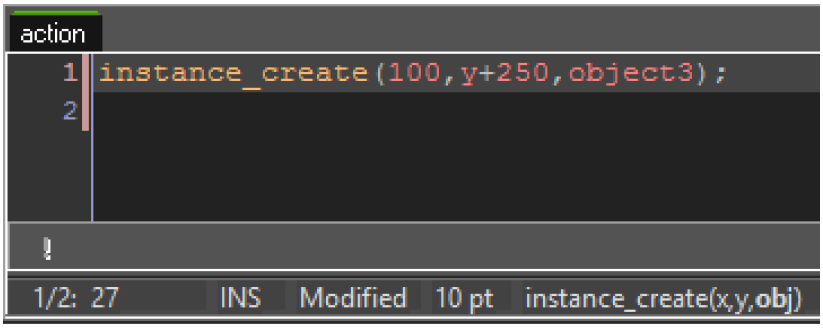
Slika 12: primjer deklaracija varijabli

2.3.2. Funkcije

GML sadrži mnogo zadanih funkcija. Zadane funkcije GML-a mogu se svrstati u više kategorija od kojih neke služe za rad s objektima, zvukovima, instancama, varijablama, tekstom, crtanjem, itd. Ukoliko je funkcija ispravno napisana biti će žute boje.

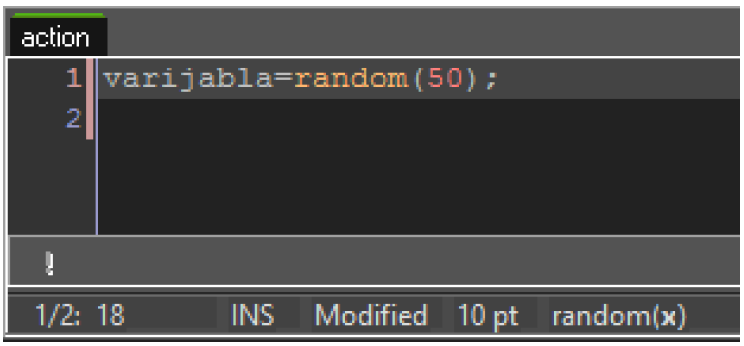
Primjeri GML funkcija:

-**Instance_create** (slika: 13) primjer je funkcije koja radi s instancama i prima argumente. Instance_create je funkcija za kreiranje instanci objekata. Kao argumente prima koordinate položaja nove instance i naziv željenog objekta. Funkcija iz primjera na slici kreira objekt „object3“, 100 piksela desno od početne točke sobe i 250 piksela ispod objekta koji poziva funkciju („y“ je programska instance varijabla objekta koji poziva funkciju).



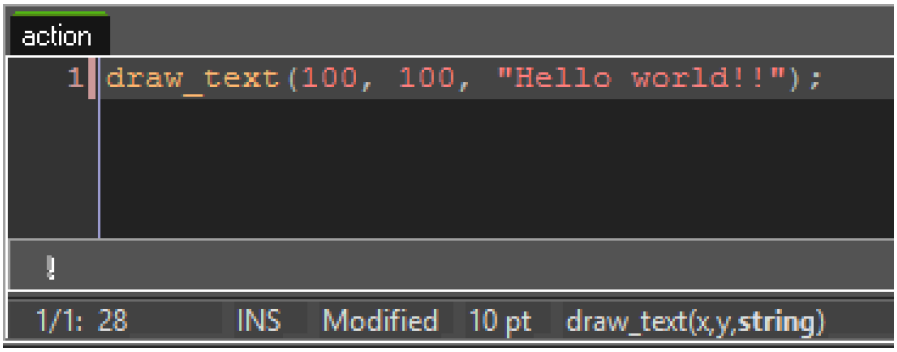
Slika 13: instance_create(x,y,obj)

-**Random** (slika: 14) je matematička funkcija koja vraća nasumični broj između 0 i vrijednosti unesenog argumenta (n). U primjeru sa slike, varijabli „varijabla“ dodjeljuje se vrijednost funkcije, koja može biti bilo koji broj između 0 i 49. Vjerojatnost za bilo koji broj jednaka je (n), odnosno 50 u ovom primjeru.



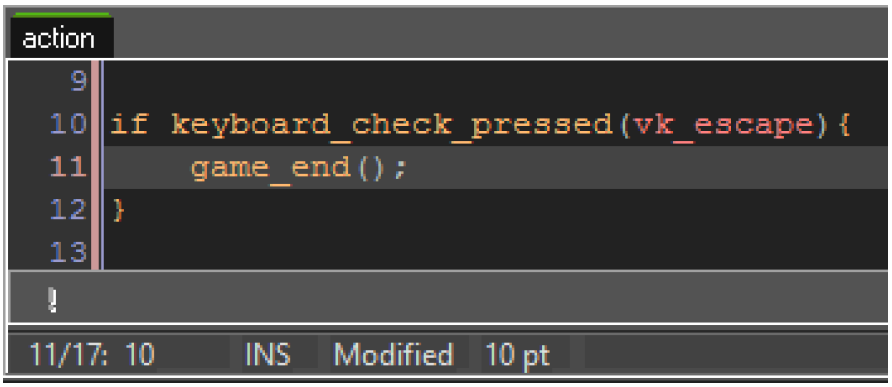
Slika 14: random(n)

-**Draw_text** (slika: 15) je funkcija za ispis teksta koja prima dva argumenta, koordinate „x“ i „y“, koje označavaju željenu poziciju teksta i jedan string argument u koji se upisuje željeni tekst za ispis. Moraju biti unutar „draw event“-a kako bi radile. Sve funkcije za crtanja i pisanje su funkcije „draw“ kategorije.



Slika 15: draw_text(x,y,string)

-**Game_end** je primjer funkcije koja ne prima ni jedan argument. „Game_end“ funkcija služi za izlazak iz igre. Slika (16), prikazuje pozivanje funkcije „game_end“ ukoliko funkcija za provjeru tipke „escape“ vrati pozitivan rezultat.



Slika 16: game_end

3.1. Opis igre

Vojnik je glavni lik igrice kojim upravlja igrač. Na samom početku, igrač se iskrcava iz transportnog helikoptera. Igra počinje u pustinji na mjestu iskrcaja. Cilj igre skupiti je što veći broj bodova, pobijediti neprijatelja i stići do obale gdje će se igrač ukrcati nazad u transportni helikopter. Neprijatelji su pripadnici neodređene terorističke organizacije koji napadaju, to jest pucaju na igrača pokušavajući ga sprječiti da obavi svoj zadatak. Prolazeći kroz igru, igrač će hodati pustinjom, neprijateljskim bazama i skloništima, susrest će vremenske neprilike i civile zatočene od strane neprijatelja. Igrač će osim neprijateljskog teritorija proći i kroz prijateljski teritorij te oslobođeni teritorij. Bodove se skuplja različitim aktivnostima: pogađanjem eksplozivnih objekata (mina ili bačvi s benzinom), pobjeđivanjem neprijatelja kao i prelaskom razina. Igrač gubi bodove ukoliko ozlijedi nenaoružanog civila. Razina je prijeđena kad igrač „živ“ izađe iz jedne sobe i prijeđe u drugu. Iz soba se najčešće izlazi tako što igrač jednostavno izađe na desnu stranu ekrana nakon što se obranio od neprijatelja. Iznimka je prva razina gdje igrač „uskoči“ u neprijateljski bunker koji je predstavlja sljedeću razinu. Igrač počinje igru sa sto posto zdravlja i pancir prsluka. Igra nije gotova dok god vojnik ima više od 0 posto zdravlja. Pancirka smanjuje štetu koju igraču nanosi neprijateljska vatra. Kad neprijatelj pogodi igrača, igrač gubi 10 posto zdravlja bez pancir prsluka, a u pancir prsluku 5 posto zdravlja i 10 posto pancir prsluka. Igrač može stradati i ukoliko stane na minu koja mu oduzme 70 posto zdravlja u slučaju da nema pancir prsluk, no ukoliko ga ima, oduzme mu se pancir prsluk i postotak zdravlja za 2 puta manji od ukupnog postotka pancir prsluka, s time da mu se maksimalno može oduzeti 60 posto pancir prsluka koja ga spasi za 30 posto zdravlja. Najopasnija situacija za igrača je da ga pregazi neprijateljsko vozilo koje mu bez obzira na pancir prsluk oduzima 3 posto zdravlja svaki trenutak u igri što je jednako 180 posto u sekundi. Na mnogim lokacijama u igri, igrač ima priliku povećati si postotak zdravlja i pancir prsluka. Igrač može s poda pokupiti medicinsku torbu koja mu obnovi 30 posto zdravlja ili novi pancirni prsluk koji opet vrijedi 100 posto bez obzira na to koliko je igrač imao u tome trenutku. Igrač može izbjeći neprijateljsku vatru saginjanjem, zaklanjanjem iza objekata, skakanjem, i povlačenjem. Može pucati na neprijatelje iz daljine prije nego ga primijete te ih ubiti prije nego stignu uzvratiti. Također ukoliko je igrač sagnut i zaklonjen poželjno mu je čekati dok se neprijatelju ne potroše metci u spremniku kako bi ga mogao napasti

dok mijenja spremnike. Igrač osim direktnog pucanja u neprijatelje može pucati u mine i bačve s benzinom koje ubiju neprijatelja u blizini. Igrač nakon pokretanja igre može odabrati različite opcije. Prva opcija je „New game“, odabravši tu opciju na zaslonu se pokreče početna animacija nakon koje započinje prva razina. „Score“ je jednak nuli, „Health“ i „Armor“ 100%, a „Level“ je 1_1. Druga opcija je „Load game“ gdje igrač bira između „Continue“ i „Select level“. „Continue“ postavlja igrača u razinu iz koje je izašao kad je zadnji put igrao uz jednake vrijednosti „Score“-a, „Health“-a i „Armor“-a. U „Select level“ opciji igrač ima mogućnost odabrati započeti igru u nekoj od „otključanih“ razina. Razina se otključa prvi put kad je igrač dostigne. Treća i zadnja opcija je „Options“ gdje igrač može odabrati koje tipke želi odabrati za upravljanje igrom te želi li igrati sa uključenim ili isključenim zvukom.

3.1.1. Tijek igre

Igra se sastoji od tri razine. Prva i treća razina igre sastoje se od četiri sobe, dok se druga razina sastoji od tri.

Razina: 1. Prva razina započinje u nultoj sobi sa animacijom transportnog helikoptera koji iskrcava igrača ispod brežuljka s napuštenom kućom. Počinje prva soba gdje igrač dobija kontrolu nad igrom. Tu se susreće s prvom minom i prvim neprijateljima. Penje se na drugi brežuljak na kojem se nalazi još nekolicina neprijatelja. Skačući s brežuljka ima priliku pokupiti prvu medicinsku torbu i pancir prsluk. Nailazi na neprijatelja. Sljedeće, dolazi do prve zastave „terorističke organizacije“ na koplju koju ima priliku zamjeniti koristeći tipku „F“ te tako doći do dodatnih bodova. Hodajući kroz pustinju nailazi na još nekoliko neprijatelja. Pred kraj prve sobe nailazi na veliku „rupu“ u podu koju je nemoguće preskočiti te u nju mora upasti ukoliko želi nastaviti igru. Upadanjem u „rupu“, upada u neprijateljsko sklonište koje je druga soba igre. U neprijateljskom skloništu nailazi na spremne neprijatelje i nekoliko zatočenih civila. Kako bi igrač izbjegao ozljeđivanje civila, mora ih preskočiti te se kratko izložiti neprijateljskoj vatri prije nego dobije priliku sagnuti se. Nakon što prođe neprijatelje na ulazu, spušta se u nižu razinu bunkera gdje ga dočekuje još neprijatelja, ali i medicinska torba i pancir prsluk. U tome trenutku igrač mora primijetiti eksplozivnu napravu na zidu koju je potrebno upucati kako bi se aktivirala i

srušila zid iza sebe kroz koji bi igrač mogao nastaviti dalje. Nastavivši, igrač se susreće s nekolicinom neprijatelja i ponovno dolazi do zatvorenog zida koji može srušiti sličnom eksplozivnom napravom. Nakon što sruši zadnji zid bunkera, izlazi van iz druge sobe i dolazi opet u pustinju. Pustinjom počinje treća soba koja je zadnja soba prva razine. Igrač silazi s ruba uništenog zida i nailazi na nove neprijatelje i mine. Nakon što prođe dalje pojavljuje se druga zastava na koplju koja mu daje mogućnost da zaradi dodatne bodove. Dalje, igrač dolazi do napuštene kuće i prehoda preko terase, nakon čega nailazi na ravninu s više praznih bačvi benzina. Te bačve dobro je ne oštetiti prije nego im se približi neprijateljsko vozilo koje se pojavi s desne strane ekrana. Neprijateljsko vozilo predstavlja prvog „boss“-a u igrici. U vozilu se nalaze dva neprijateljska vojnika koji pucaju na igrača. Igračev cilj uništiti je vozilo prije nego ga pregazi ili potjera skroz do zida kuće. Zid je previsok da bi se igrač popeo nazad. Nakon što igrač pobjedi vozilo treba nastaviti hodati u desno dok ne izađe iz sobe i time prijeđe prvu razinu.

Razina: 2. Druga razina počinje u pustinji na prijateljskom teritoriju. Igrač nailazi na civile koji hodaju po terasi uništene kuće. Oni za igrača imaju spremljenu medicinsku torbu i pancirni prsluk. Iza kuće na koplju stoji prijateljska zastava koju igrač nema mogućnost zamjeniti svojom. Igrač se dalje penje na brežuljak koji je zarastao u raslinje te je pun mina i time prelazi granicu u neprijateljski teritorij gdje opet nailazi na neprijatelje. Nailazi na olupinu od automobila koja nema svrhu osim dekorativne, a iza auta nalazi se zatvorena zgrada neprijateljske baze. Na zidu stoji eksplozivna naprava koju igrač može aktivirati kako bi ušao u tu zgradu. Iza samog ulaza u zgradu dočekuje ga još jedan neprijatelj. Peta soba interijer je neprijateljske baze u kojoj igrača neprijatelji dočekaju sa više strana. Igrač treba nastaviti samo ravno kako bi izašao iz baze. Na kraju baze nailazi ponovno na zid koji je potrebno srušiti. Nakon što ga sruši potrebno je samo proći kroz „rupu“ u zidu i tako igrač izlazi iz pete sobe. Ulaskom u šestu sobu, igrač nailazi na oslobođeni teritorij sa svojim zastavama i vojnicima. Nakon što im igrač priđe, oni uočavaju nepriijateljsko vozilo i otvaraju vatru. Igrač im mora pomoći kako bi ga uspješno uništili prije nego ih pregazi. Igrač zatim treba samo nastaviti kroz pustinju i tako preći drugu razinu.

Razina: 3. Hodajući kroz pustinju u sedmoj sobi igrač nailazi na neprijatelje i pustinjsku oluju. Pustinjska oluja znatno otežava vidljivost te neprijatelje i mine čini teže vidljivima što zahtjeva veću opreznost igrača. Na otprilike sredini sobe nalazi se

neprijateljska zastava na koplju koju igrač opet može zamjeniti za dodatne bodove. Igrač se penje na brežuljak gdje ga dočekuje nekoliko neprijatelja pokraj parkiranog vozila koje ne predstavlja opasnost. Kraj sedme sobe obilježavaju otvorena vrata glavnog skrovišta neprijatelja kroz koja igrač prolazi bez poteškoća. Ulaskom u glavno neprijateljsko skrovište, osmu sobu, igrač nailazi na teretno dizalo kojim se potrebno spustiti do najniže razine skloništa. Na gornjoj, 4. razini skloništa igrač ima priliku pokupiti medicinsku torbu i pancirku. Na trećoj razini nalazi se jedan neprijateljski vojnik, na drugoj jedan neprijatelj i dvoje zarobljenih civila, na prvoj jedan neprijatelj i medicinska torba, a na nultoj, najnižoj razini, po jedan neprijatelj sa svake strane. Igračev cilj je skrenuti desno i proći u sljedeću sobu. Ulaskom u devetu sobu, igrača dočekuje puno neprijatelja. Nakon toga, igrač mora srušiti zid kako bi se mogao spustiti na još nižu razinu. Na toj razini dočekuje ga još nekoliko neprijatelja. Kada ih pobijedi, treba skočiti na posljednju razina skloništa. Posljednja razina skloništa je i neprijateljsko sjedište gdje igrač neprijatelje prekida u slavlju. Nakon što ih sve pobjedi, igrač napušta glavni bunker i dolazi nazad na površinu. Deveta soba je pustinja, tu igrač nailazi na puno neprijatelja i na oklopno vozilo.

3.2. O implementaciji

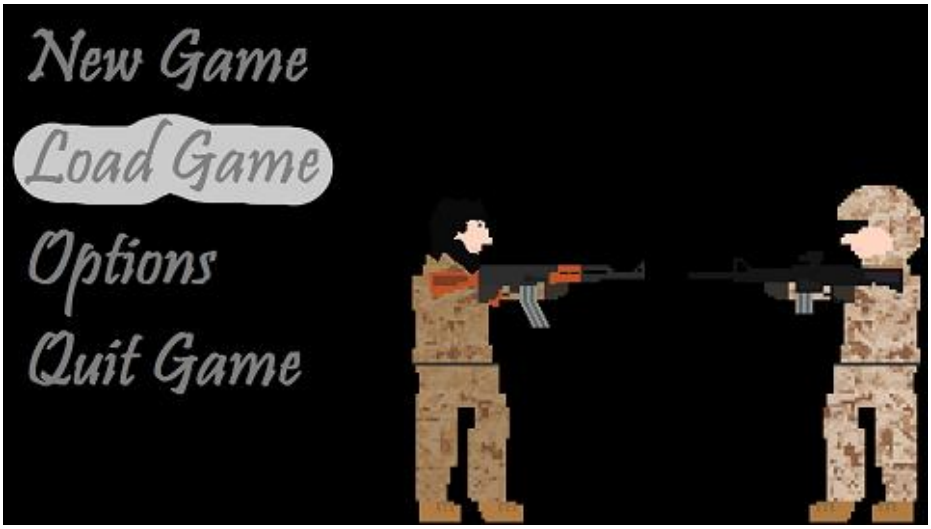
Za izradu igre korišteno je preko 120 game-maker objekata, 130 sprite-ova, 170 slika, 8 zvukova, 17 soba i 6 fontova. U ovome poglavlju biti će objašnjeni neki objekti, njihovi događaji, radnje i neki od resursa koje navedeni objekti koriste.

3.2.1. Introroom i pripadajući objekti

Nakon pokretanja igre, otvara se prva soba, introroom. Introroom sadrži sedam objekata te predstavlja glavni izbornik. U kojem igrač može birati između „New Game“, „Load Game“, „Options“ i „Quit Game“. Prva naredba koju introroom izvršava nalazi se u creation code-u sobe te glasi: „window_set_fullscreen(true);“. Ta naredba postavlja širinu i visinu ekrana na „fullscreen“. Sljedeće, introroom kreira dva objekta: „ob_menu“ i „ob_startigre_zvuk“. „Startigre_zvuk“ objekt ima samo jedan event, „create event“. U „create event“-u provjerava postoji li „controls.sav“ datoteka te ukoliko postoji, učitava ga i sačuvanu vrijednost upisuje u varijablu „global.zvuk1“. Ukoliko je vrijednost te varijable jednaka 1, pojaviti će se zvukovi u igrici. Tu vrijednost igrač može promjeniti. Većina objekata koji se mogu pojaviti u ovoj sobi funkcioniraju na istom principu.

3.2.1.1. ob_menu

Drugi objekt koji introroom kreira je „ob_menu“. Tim objektom igrač može upravljati. „Ob menu“ služi za odabir između pokretanja nove igre „New game“, otvaranje daljnjih izbornika: „Load game“ („ob_load“ 3.2.1.2.) i „Options“ („ob_opt“ 3.2.1.3.) te izlaženja iz igre „Quit game“. Prvi „ob_menu“ event je „create event“. Prva naredba „create event“-a je dodjeljivanje vrijednosti 1 varijabli „menju“. **Varijabla „menju“** označava različite odabrane mogućnosti u glavnom izborniku. Zadana vrijednost 1 označava „New Game“, vrijednost 2 „Load Game“, vrijednost 3 „Options“ i vrijednost 4 „Quit Game“ (slika: 17).



Slika 17: ob_menu

Igrač tipkama „up“ i „down“ mijenja vrijednost te varijable kao i odabranu mogućnost u izborniku. Pritisnuvši tipku „down“, igrač vrijednost varijable „menju“ povećava za jedan i tako nova odabrana mogućnost postaje ona za jedno mjesto ispod prethodno odabrane. Ukoliko „menju“ dosegne vrijednost za jedan veću od maksimalne vrijednosti priključene jednoj od mogućnosti, dodjeljuje joj se najmanja koja je priključena nekoj mogućnosti. Pritiskom na tipku „up“ postiže se suprotno. Obje te tipke osim pomicanja kroz mogućnosti emitiraju zvuk „sd_bang“ zbog dodatnog efekta, ukoliko je vrijednost varijable jednaka 1. Druga naredba „create event“-a dodjeljuje vrijednost 0 varijabli „global.loadnatpis“. Više detalja o toj varijabli nalazi se u nastavku odlomka. „Create event“ otvara dvije datoteke: „controls.sav“ i „leval.sav“. Objekt igrača, koji je objašnjen u sljedećem odlomku, otvara „controls.sav“ te prema njemu određuje koje tipke igrač treba koristiti za upravljanje igrom. Prije otvaranja “controls.sav“ provjerava se postoji li već, ukoliko NE postoji, dodjeljuje mu se zadane vrijednosti, dakle zadane tipke za upravljanje. „Leval.sav“ se također otvara ukoliko NE postoji te mu se upisuje zadana vrijednost 1. „Leval.sav“ služi za „Select level“ opciju kojom igrač može započeti igru iz neke od razina koje je ranije dostigao, a ako „save file“ ne postoji igrač, može igru započeti samo u prvoj razini.

„Step event“ provjerava vrijednost varijable „menju“ te prema njoj dodjeljuje potrebni sprite objektu (slika: 18). Ukoliko varijabla „menju“ dobije vrijednost veću od koliko ima mogućnosti dodjeljuje joj se najmanja vrijednost i obrnuto.

```
if(menju==1){
    sprite_index = sp_menu_1;
    image_speed = 0.2;
}
```

Slika 18: Primjer: "menju==1"

„Press enter event“. Pritiskom na tipku enter, pokreće se odabrana radnja. Jedina provjera koju ovaj event provjerava je vrijednost varijable „menju“ prema kojoj djeluje. Ukoliko je vrijednost jednaka 1, otvara se soba 1_0 i vrijednosti -50 pridružuje se „varijabli global.heltdalje“. Tu varijablu koriste objekti „ob_player“ (3.2.2.) i „ob_player_ghost“ (3.2.3.). U slučaju da je vrijednost jednaka 2, stvara se „ob_load“ objekt i uništava „ob_menu“. Slučaj vrijednosti 3 je jednak slučaju 2, osim što se stvara objekt „ob_opt“ (3.2.1.3.). Vrijednost 4 zatvara igru.

3.2.1.2. ob_load

„Ob_load“ je poput „ob_menu“ objekt izbornika te kao takav funkcionira veoma slično objektu „ob_menu“. „Ob_load“ nudi tri mogućnosti. Nastavka igre na mjestu gdje je igrač stao kad je zadnji put igrao igru - „Continue“, ulazak u daljnji izbornik „Select levels“ („ob_sellvl“ 3.2.1.4.) i povratak u prošli izbornik „Back“ koji vraća objekt „ob_menu“ (slika: 19). U „create event“-u „ob_load“ definira samo jednu varijablu, varijablu „loadd“. Ta varijabla radi na isti način kao varijabla „menju“ objekta „ob_menu“ (opisana u 4.2.1.1.). „Step event“ provjerava vrijednosti „loadd“-a te prema njemu mijenja spriteove objekta. Pritiskom na tipku enter, provjerava se „loadd“ varijabla. U slučaju da je jednaka 1, program provjerava postoji li „Save.sav“ datoteka. Ona kreira objekt „ob_player“ ulaskom u novu sobu i u njega sprema id sobe, razinu zdravlja i pancirke te količinu prikupljenih bodova. Ukoliko datoteka postoji i sadrži ID bilo koje sobe osim prve, otvara se zadnja prethodno igrana soba sa svim parametrima koje „save file“ sadrži. Ako ta datoteka ne postoji ili je učitan id prve sobe u igri, stvara se jednostavni objekt „ob_nemaload“ (slika: 20). **„Ob_nemaload“** na zaslonu ispisuje „Nothing has been saved“, a uklanja se u trenutku kad igrač promjeni vrijednost „loadd“, dakle ukoliko stisne tipke „up“ ili „down“. Ako je vrijednost „loadd“ varijable jednaka 2, stvara se „ob_sellvl“ (3.2.1.4.) i uništava „ob_load“, a ukoliko je jednaka 3, također se uništava trenutni objekt „ob_load“, no kreira se „ob_menu“, to jest igrač se vraća na prethodnu stranicu.

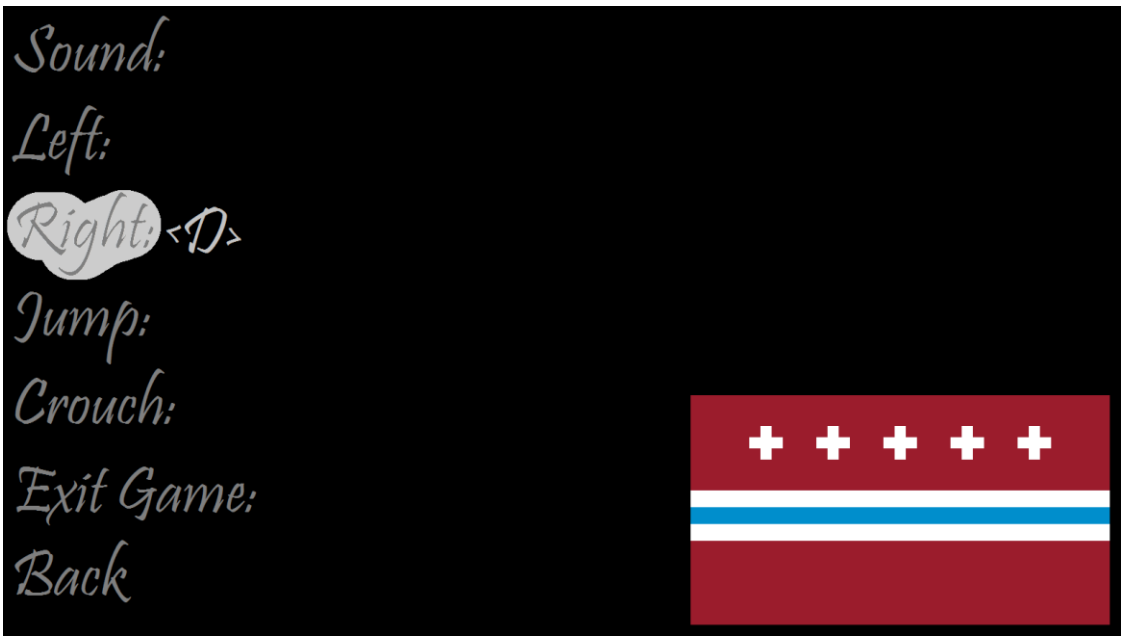
Slika 19: ob_load

Slika 20: ob_nema_load

3.2.1.3. „ob_opt“ i „ob_swich“

Objekti „ob_opt“ i „ob_swich“ dijele zajedničko sučelje. „Opt“ je skraćeno od options te služi kao sučelje za odabir kontrola u igri i odabir opcije isključenog ili uključenog zvuka. „Ob_swich“ služi za prikaz odabrane opcije te mijenjanje mogućnosti. Prikazan je kao neka od mogućnosti (primjer sa slike 21 ob_swich == <D>). U „create event“-u „ob_opt“ definira varijablu „opti“ i kreira objekt „ob_swich“. „Opti“ služi istoj svrsi kao i varijabla „menju“ (opisana u 4.2.1.1.) objekta „ob_menu“. Osim toga, „opti“ izmjenjuje i spriteove kreiranoga „ob_swich“. U „ob_opt“ mijenja se označeni redak opcije, a u „ob_swich“ simbol slova i redak (koordinate) u kojem se nalazi. Sprite „ob_swich“ mijenja se i tipkama „left“ i „right“. Prema tome, u „ob_opt“ odabire se radnja u igri čije kontrole igrač želi promjeniti, a „ob_swich“ prikazuje mogućnosti, to jest „alternative“ tog retka. Tipkama „up“ i „down“ bira se retke, a tipkama „left“ i „right“ mogućnosti tih redaka. „Ob_swich“ u „create event-u“ učitava „controls.sav“ datoteku iz koje definira „opto1,...., opto6“ varijable u koje se upisuju odabrane kontrole. Kako se koristi te varijable u igri detaljnije je opisano u odlomku „step event“-a objekta „ob_player“ (3.2.2.). Tipkama „left“ i „right“ igrač izmjenjuje vrijednosti tih varijabli ovisno o retku koji je označen. Redak se označava gore

spomenutom varijablom „opti“. Kada igrač promijeni vrijednost optox varijable u „step event“-u objekta „ob_swich“, izmjenjuje se aktivni sprite. Slika 21 prikazuje situaciju u kojoj je vrijednost „opto“ varijable objekta „ob_opt“ jednaka 3 te je zbog toga označen treći redak „Right:“, a objekt „ob_swich“ je prikazan pored njega. Iz slike se također vidi da je vrijednost varijable „opto3“ jednaka dva te zbog toga objekt „ob_swich“ prikazuje drugu po redu mogućnost „<D>“. S ovom odabranom opcijom kako bi se kretao u desno igrač mora koristiti tipku „d“.



Slika 21: ob_opt i ob_swich

3.2.1.4. ob_sellvl

Ovaj objekt predstavlja sučelje koje služi za odabir razine iz koje igrač želi započeti igru (slika: 22). Do ovog sučelja dolazi preko „ob_load“ objekta. Igrač može listati postojeće razine te odabrati samo one koje je prethodno „otključao“. Razina se otključa u trenutku kad je prvi put igrač otvori u igri. U „create event“-u definira varijablu „lev“ i dodjeljuje joj vrijednost 1. Varijabla „lev“ služi za „kretanje“ kroz objekt. Igrač tipkama „left“ i „right“ povećava ili smanjuje vrijednost te varijable, a u „step event“-u se provjerava ta varijabla i prema njenoj vrijednosti prikazuje odgovarajući sprite. U „create event“-u se zatim provjerava postoji li „leval.sav“ datoteka. Ona pohranjuje jednu varijablu. Svaki puta kad igrač uđe u neku novu sobu, provjerava se kolika je vrijednost te varijable i uspoređuje se s vrijednošću koju nudi soba. Ukoliko „leval.sav“ ima manju vrijednost, dodjeljuje mu se nova. Time igrač, kad god uđe u

novu sobu, razinu, „otključa“ novu u „Select level“ opciji. Ukoliko „leval.sav“ postoji, čita se sačuvana vrijednost i stavlja se u novu varijablu „leval“. Ukoliko ne postoji, varijabli „leval“ dodjeljuje se vrijednost 1. Postoji deset različitih soba za odabrati u ovoj opciji te zbog toga varijabla „lev“ može poprimiti samo vrijednosti između 1 i 10. Ukoliko vrijednosti dođe do 10 i igrač nastavi pritiskati tipku „right“ za povećevanje vrijednosti „lev“, u „step event-u“ vrijednost će postaviti nazad na 10. Isto vrijedi i za tipku „left“ i vrijednost 1. „Step event“ provjerava vrijednost „lev“ varijable, a zatim i „leval“ varijable. Prema „lev“ određuje razinu koju igrač želi prikazati, a nakon provjere „leval“ određuje je li ta razina otključana. Ukoliko je otključana, prikazuje je se sprite te razine u boji, a ukoliko nije otključana, odabrana razina se prikazuje crno-bijelo. Pritiskom na tipku „enter“ igrač odabire željenu sobu, a program započinje provjeru „lev“ varijable, odnosno provjerava koja je soba odabrana. Nakon toga se provjerava koji je sprite aktivan. Ako je sprite u boji, pokreće se ta soba i global varijabli „global.heltdalje“ pridružuje se vrijednost -50, a ako je crno-bijeli sprite, ne događa se ništa. „Global.heltdalje“ varijabla definira zdravlje i pancirku u igri te ukupan broj bodova. Ukoliko „global.heltdalje“ dobije vrijednost -50, u igri se vrijednosti postavljaju na početnu vrijednost. Više o toj varijabli je u odlomku (3.2.3.), o objektu „ob_player_ghost“.

Slika 22: ob_sellvl. "Otključana" razina 2_2

3.2.2. ob_player

Objekt „ob_player“ ima dubinu 1 i pridruženi sprite sp_player (slika 23). Inicijalna svrha ovog objekta bila je da služi kao maska za „ob_player_ ghost“, no kasnije su mu dodane još neke različite funkcije. „Ob_player“ je glavni objekt igrača kojim se kreće kroz igru koristeći tipkovnicu. Objekt ob_player prati jedini view u igrici View0 (2.3.2.). Također ga prati i „ob_player_ghost“ (3.2.3.) koji je zadužen za izgled igrača. Prati ga tako što u step eventu izjednačava svoje varijable pozicije s pozicijom „ob_player“-a:

```
ob_player_ghost.x= ob_player.x;  ob_player_ghost.y= ob_player.y;
```

Slika 23: sp_player

Sprite „sp_player“ ima nacrtane krajnje točke koje služe za kretanje kroz igricu. Dakle, „ob_player“ se može kretati prema solid objektu dok mu se ne primakne s jednom od tih točaka. Crveno slovo „S“ služi samo kako bi bio uočljiv prilikom kreiranja soba te nema veze sa game-playem.

Varijabla „ccc“ označava poziciju klečanja ili stajanja igrača. Moguće vrijednosti su 0 i 1. Vrijednost 0 označava da igrač stoji, a vrijednost 1 označava klečanje. Vrijednost 1 sprječava igrača da se kreće horizontalno i vertikalno, odnosno da hoda i skače. Stvaranje objekta metka „ob_bullet“ također ovisi o ovoj varijabli zbog različite pozicije puške. U step eventu vrijednost „ccc“ se mijenja pritiskom na tipke „c“ ili „s“ (ovisno o tome koje je odabrana u opcijama), odnosno otpuštanjem istih.

Varijabla „magazin“ predstavlja broj preostalih metaka u spremniku. Moguće vrijednosti su između 0 i 30. Za svaki ispaljeni metak smanjuje se za 1, a kad dođe do vrijednosti 0 stvara objekt „ob_magazin“ koji pada i ostaje na podu, te kratko sprječava ob_player-a da stvara nove metke predstavljajući zamjenu magazina. Na kraju vraća vrijednost na početnu vrijednost, 30.

Varijabla vrste global „global.alive“ ima vrijednost 1 sve dok je igrač „živ“. Vrijednosti se mijenja u 0 tek kada health varijabla „hpp“ (3.2.3.) objekta „ob_player_ghost“ dođe do nule ili manje, odnosno kada igrač izgubi health, to jest kad umre. Kad ova varijabla dobije vrijednost nule, igrač gubi sve mogućnosti kretanja, pucanja i sl. Još promjena događa se istovremeno kad se i vrijednost ove varijable postane 0, to je detaljnije opisano u potpoglavlju 4.2.2.2.

„Friction“ je programska varijabla. Kroz cijelu igru je definirana na 0.5 te se ne mijenja. U svakom trenutku igre smanjuje brzinu „ob_player“-a za 0.5.

3.2.2.1. Create Event

Prvi event je create_event koji definira varijable „ccc“ na 0, „magazin“ na 30, varijablu vrste global „global.alive“ na 1 i varijablu friction na 0.5.

Zatim provjerava postoji li datoteka „controls.sav“ te ako postoji, pokreće ga za čitanje i sprema string u varijablu „loadfile“ iz koje se riječ po riječ varijable opto1,2....6 učitavaju. Na kraju se datoteka zatvara (slika: 24).

```
if(file_exists("controls.sav")){

        loadfile= file_text_open_read("controls.sav");
        opto1= file_text_read_real(loadfile);
        opto2= file_text_read_real(loadfile);
        opto3= file_text_read_real(loadfile);
        opto4= file_text_read_real(loadfile);
        opto5= file_text_read_real(loadfile);
        opto6= file_text_read_real(loadfile);
        file_text_close(loadfile);
        }
```

Slika 24: Učitavanje kontrola

Varijable opto1,2...6 izmjenjuje i sprema objekt „ob_swich“. Te varijable služe za izmjenjivanje kontrola nad objektom igrača „ob_player“ i biranje zvuka. Sve optoX varijable mogu sadržavati vrijednosti 1 ili 2. Varijabla „opto1“ služi za upravljanje zvukovima. Ako je njezina vrijednost 1 zvukovi rade (slika: 25).

```
if(opto1==1){
audio_play_sound(sd_bang, 10, false);}
```

Slika 25: Ovisnost zvuka o „opto1“

O „opto2“ ovisi hoće li igrač za kretanje lijevo koristiti tipku „left“ ili „a“ (slika: 26), o „opto3“ hoće li za kretanje desno koristiti tipku „right“ ili „d“. „Opto4“ definira skakanje tipkama „up“ ili „w“. „Opto5“ definira klečanje/ saginjanje tipkama „c“ ili „s“, a „opto6“ hoće li igrač za izlazak iz igre, odnosno povratak u „introroom“, koristiti tipku „escape“ ili „F4“.

```
if(opto2==1){
if(keyboard_check_direct(vk_left)){
    if(ccc!=1){
    hspeed=-6;
}
    image_xscale=-1;

}
}
else if(opto2==2){
if(keyboard_check_direct(ord('A'))){
    if(ccc!=1){
    hspeed=-6;
}
    image_xscale=-1;
}
}
```

Slika 26: Kretanje lijevo

Varijabla „hspeed“ programska je varijabla. Ona definira kreće li se zadani objekt horizontalno. Zadana vrijednost je 0 što označava stajanje. Vrijednost 1 pokreće objekt za jedan piksel desno svaki trenutak, a vrijednost -1 lijevo.

Varijabla „vspeed“ slična je varijabli „hspeed“. Ona je programska je varijabla zadane vrijednosti 0. Razlikuje se po tome što definira kretanje objekta vertikalno. Vrijednost 1 pokreće objekt za jedan piksel „dolje“, a vrijednost -1 „gore“.

Varijabla „image_xscale“ također je programska varijabla. Služi sa rastezanje slike prema x osi. Zadana vrijednost je jednaka 1 što znači da se slika prikazuje onakvom kakva jest. Vrijednost 3 rastegnula bi sliku za 3 puta. U projektu, ta je varijabla najčešće korištena za usmjeravanje objekta lijevo, odnosno desno uz izmjenjivanje vrijednosti na -1, odnosno 1.

3.2.2.2. Step Event

Step event, svaki trenutak igre, kreće tako što prvo definira programsku varijablu „vspeed“ te joj dodjeljuje vrijednost +=1.1, odnosno igrača ubrzava prema „dolje“ za 1.1 svaki trenutak. „Vspeed“ u ovom slučaju predstavlja gravitaciju.

Sljedeće se otvara execute code action koji provjerava varijablu „global.alive“, odnosno provjerava je li igrač „živ“ te može li izvršavati sljedeće radnje.

Sljedeće provjerava varijablu „opto2“ te ovisno o njoj provjerava tipke „a“ ili „left“. Ukoliko je neka od tih tipki pritisnuta, provjerava kleči li igrač (varijabla „ccc“), te ako ne kleči, „hspeed“ varijabli dodjeljuje vrijednost -6. Ukoliko igrač ne stoji, ne može hodati te „hspeed“ ne mijenja vrijednost. Vrijednost „image_xscale“ postaje -1 što znači da se sprite zrcali i gleda u drugu stranu.

Zatim dolazi slična radnja, provjerava se „opto3“ te ovisno o njoj tipke „d“ ili „right“. Igrač se kreće u desno ako je neka od te dvije tipke pritisnuta i ako igrač ne kleči tako što se varijabli „hpseed“ daje vrijednost 6, a varijabli „image_xscale“ daje se vrijednost 1.

Slijedi provjera „opto5“. Ukoliko je vrijednost 1, provjerava se tipka „c“, a za vrijednost 2, tipka „s“. Pritiskom na neku od tih tipki vrijednost varijable „ccc“ postaje 1, odnosno igrač klekne te mu se time između ostaloga oduzme mogućnost kretanja. Puštanjem pritisnute tipke, vrijednost varijable „ccc“ postaje 0 te igrač ustane. Mogućnost klečanja prvenstveno služi kako bi se izbjeglo neprijateljsku vatru.

Provjerom varijable „opto4“ provjerava se tipka za skakanje. Ukoliko je vrijednost „opto4“ 1 ili 2, igrač skače tipkom „up“, odnosno „w“ tim redom. Pritiskom na neku od tih tipki provjerava se kleči li igrač i funkcija place_free(x,y+2). Ta funkcija daje pozitivan rezultat ako se solid objekt nalazi na unesenoj koordinati u odnosu na objekt čija je funkcija. U ovome slučaju, place_free funkcija traži solid objekt dva piksela ispod objekta ob_player. Dakle, igrač može skočiti ukoliko stoji na nekome solid objektu. Time je sprječeno duplo skakanje. Skakanje se izvršava tako što se igraču vrijednost „vspeed“ varijable promjeni na -17 te ga „gravitacija“ zadana na početku step eventa vrati na solid objekt što će kasnije biti detaljnije pojašnjeno.

Pritiskom na tipku „space“ za „pucanje“, okida se alarm[0] koji je detaljno objašnjen u potpoglavlju (3.2.2.3).

Slijedi „opto6“ provjera koja provjerava tipku potrebnu za izlazak iz igre. Ako je vrijednost „opto6“ jednaka 1, provjerava se tipka „F4“, a ako je vrijednost jednaka 2, provjerava se tipka „escape“. Pritiskom na jednu od te dvije tipke, izlazi se iz „igre“ te se sučelje prebacuje u „introroom“ i pokreće se „room end“ event.

Zadnja provjera ovog execute code actiona je „hpp“ varijabla objekta „ob_player_ghost“ koja je detaljnije objašnjena u potpoglavlju (3.2.3.). Ovdje se samo provjerava je li ta varijabla jednaka ili manja od nule. Ukoliko jest, znači da igrač umire. Varijabli „global.alive“ vrijednost se mijenja u 0. Sprite se mijenja i kod „ob_player“-a i kod „ob_player_ghost“-a u „sp_player_ghost_ded“ (slika: 27). Stvara se objekt „ob_ded“ (3.2.5.) koji mijenja pozadinu u crvenu boja kako bi se naglasio kraj igranja te se alarm[10] postavlja na 100, što je ~1,667 sekundi. Alarm[10] vraća sučelje na početnu stranu igrice „introroom“i aktivira „room_end“ event.

Slika 27: sp_player_ghost_ded

Nakon execute code actiona obavljaju se dvije slične provjere, provjere kolizije vertikalne i horizontalne. Tu je korištena „drag and drop“ metoda umjesto kodiranja kako bi provjere bile uočljivije zbog njihove važnosti. Ove iste provjere, osim „ob_player“-a, sadrže i svi ostali objekti koji se kreću kroz dvije prostorne dimenzije. Te dvije provjere funkcioniraju tako što za svaki „step event“ provjeravaju nalazi li se koji „solid“ objekt na putu zadanog objekta. U slučaju horizontalne provjere, provjeravaju se objekti na „hspeed“ udaljenosti od zadanog objekta na x koordinatnoj osi. Dakle, ako se, na primjer, igrač kreće brzinom 6 piksela u svakom trenutku igre (hspeed==6), svaki „step event“ provjerava sljedećih 6 piksela na svome putu. U slučaju vertikalne provjere, provjerava se „vspeed“ udaljenost na y koordinatnoj osi. Nakon tih provjera u slučaju da dolazi do susreta sa „solid“ objektom, pokreće se funkcija „move to contact“. Ta funkcija dopusti zadanom objektu da se primakne susretnutom solid objektu provjeravajući piksel po piksel. Kad se „move to contact“ izvrši, horizontalnoj odnosno vertikalnoj brzini pridružuje se vrijednost 0 te se objekt zaustavlja.

3.2.2.3. Alarmi

„Ob_player“ ima tri različita „alarm event“-a. Alarme označene brojevima 0,1 i 10. U slučaju „ob_player“-a, svi alarmi se aktiviraju unutar „step event“-a. Za aktiviranje alarma 0 i 1, igrač mora biti „živ“ i držati tipku „space“. Alarm 10 aktivira „smrt“ igrača, dakle kad health („hpp“) drugog objekta igrača „ob_player_ghost“ dosegne nulu.

Alarm[0] prvo provjerava drži li igrač još uvijek tipku za pucanje, tipku „space“, a zatim već spomenutu varijablu „magazin“. Ukoliko „magazin“ vraća vrijednost veću od nule, smanjuje mu se vrijednost za jedan, a varijabli „mecigore“ objekta „ob_text“ (3.2.4.) vrijednost se povećava za 18. Objekt „ob_text“ koji prikazuje preostale metke u spremniku na lijevoj strani ekrana time ih podiže za 18 piksela što izgleda kao da donji metak s prikaza nestaje te da ih ostaje za jedan manje. Prije stvaranja samog metka „ob_bullet“ u cijevi puške, provjerava se stoji li igrač ili kleči i na koju stranu je okrenut pa se prema tome se određuje koordinata stvaranja metka. Ako je igrač odabrao igrati sa zvukovima „opto1==1“, pokreće se zvuk „sd_bang“ koji simulira eksploziju i stvara se „ob_shell“ (3.2.5.) objekt koji predstavlja praznu čahuru metka. Stvara se „metak“ u cijevi puške, objekt „ob_bullet“, koji dalje sam upravlja svojim kretnjama i funkcijama te se alarm[0] postavlja na vrijednost pet što znači da se ponovo izvršava za pet trenutaka (~8,33 stotinke). Ali ako „magazin“ vrati vrijednost nula ili manje, „alarm[1]“ se postavlja na na 95(~1,38s) te se stvara objekt „ob_magazin“. On predstavlja prazni spremnik koji pada na pod. Prilikom stvaranja „ob_magazin“-a, provjerava se stoji li ili kleči igrač te na koju je stranu okrenut. Prema tome se određuje na kojim koordinatama u odnosu na igrača se stvara.

Alarm[1] se aktivira kada se „magazin“ isprazni. Njegova aktivacija je odgođena na ~1,38 sekundi kako bi se simuliralo vrijeme potrebno za zamjenu spremnika. Uloge ovog alarma, osim odgođenog aktiviranja, su ponovno aktiviranje alarma 0, postavljanje početne vrijednosti u varijablu „magazin“ koja je jednaka 30 i postavljanje vrijednosti 0 u varijablu „mecigore“ objekta „ob_text“.

Alarm[10] aktivira se ~1667s nakon što se potroši sav health igrača, to jest kad igrač „umre“. Jedina funkcija mu je prebacivanje zaslona iz trenutne u početnu, sobu „introom“. To ujedno i aktivira „room_end event“.

3.2.2.4. Room eventi

Postoje tri različita „room event“-a u slučaju objekta „ob_player“. „Outside room“ je event koji se aktivira nakon što „ob_player“ napusti prostor sobe. „Room start“ se aktivira prvi trenutak nakon nastanka sobe, a kad se aktivira neka od funkcija za zatvaranje sobe, aktivira se „Room end“ event. Iz sobe se izlazi koristeći tipke „F4“ ili „escape“ te ako igrač izgubi sve health bodove, aktiviranjem alarma 10.

„Outside room“ je ovom slučaju korišten kao event koji se aktivira nakon što igrač pređe razinu. Nakon aktivacije prvo provjerava postoji li sljedeća soba i ukoliko postoji, igrač prelazi tamo. Ukoliko ne postoji, otvara se početna soba („introroom“). Ta mogućnost se ne može dogoditi za vrijeme prelaženja igre. Sljedeće se izvršava dodjela vrijednosti trima global varijablama: „global.heltdalje“, „global.armodalje“, global.scordalje. Dodjeljuju im se vrijednosti varijabli objekta „ob_player_ghost“: „hpp“, „armo“ i „scor“ koje predstavljaju razinu zdravlja, ispravnosti pancirke i do sad skupljenih bodova. Razlog ovom pridruživanju vrijednosti je taj što su global varijable neovisne o sobi, dok se varijable objekata postavljaju na početnu vrijednost svakom promjenom sobe. Kako se one koriste dalje, opisano je u nastavku teksta koji govori o „room start“ eventu.

„Room start“ event počinje provjerom postoji li „Save.sav“ datoteka. Ukoliko postoji, briše se. Kad se osigura njihovo nepostojanje, kreira se novi te se u njega spremaju četiri varijable. Prva varijabla je id trenutne sobe, a preostale su varijable vrste global: „heltdalje“, „armodalje“ i „scordalje“. Razlog stvaranja „Save.sav“ datoteke je taj što omogućava igraču, koji želi ući u igru, pronalaženje opcije Load game>Continue u glavnom izborniku te nastavljanje igre s bodovima koje je prethodno zaradio. „Room start“ event zatim kreira objekt „ob_text“ koji stvara sučelje na vrhu ekrana u kojem se nalaze podaci o preostalom zdravlju, stanju pancirke bodovima i trenutnoj sobi. „Ob_text“ također prikazuje i količinu preostalih metaka u spremniku na lijevoj strani zaslona i sunce. Zadnja provjera koju vrši ovaj event je provjera radi li se o eksterijeru, to jest ako se razina odvija vani na otvorenom, stvara objekt „ob_cloud_creator“ koji stvara oblake na nebu.

„Room end“ event ima samo jednu naredbu i to je pokretanje funkcije „audio_stop_all();“ koja gasi sve zvukove koji su preostali raditi u trenutku kad igrač napušta sobu.

3.2.3. ob_player_ghost

Glavna uloga objekta „ob_player_ghost“ je izgled glavnog objekta „ob_player“. Kao što je u prošlom odlomku opisano, „ob_player_ghost“ svaki „step event“ prati „ob_player“ kojim igrač upravlja pomoću tipkovnice. „Ob_player_ghost“ prikazan je sprite-om „sp_player_ghost“ (slika: 28) te prema promjenama glavnog objekta izmjenjuje sprite-ove. Osim za izgled, „ob_player_ghost“ zadužen je i za praćenje zdravlja, pancir prsluka te ukupnih prikupljenih bodova u igri.

Slika 28: ob_player_ghost

Varijabla „global.heltdalje“ služi za spremanje zdravlja igrača u slučaju izlaženja iz soba te se pomoću nje postavlja i ostale varijable za pancirku i bodove. Ukoliko objekt „ob_player“ napušta sobu, razinu zdravlja, pancir prsluka i bodova sprema u varijable global vrste „heltdalje“, „armodalje“ i „scordalje“. Global varijable se koriste zbog toga što su neovisne o sobi, a varijable objekata se restartiraju svakom promjenom sobe. Kod prelaska u sljedeću sobu, zdravlju, pancir prsluku i bodovima pridružuje se vrijednost te tri global varijable pa ih se sprema u „save.sav“ datoteku. Razlog spremanja je mogućnost igrača da se, ukoliko izađe iz igre, može vratiti na zadnju lokaciju u igri sa zapamćenim vrijednostima varijabli uz odabir opcije „Continue“ (3.2.1.1.). Kada igrač odabere mogućnosti „new game“ ili „select level“, varijabli „heltdalje“ pridružuje se vrijednost -50. Ako je ta vrijednost jednaka -50 prilikom kreiranja objekta („create event“), „ob_player“ postavlja početne vrijednosti zdravlju, pancirki i ukupnim bodovima (100%, 100%, 0) što označava početak nove igre. „Select level“ je također početak nove igre, samo ne iz prve sobe, već one odabrane.

Varijable „hpp“, „armo“ i „scor“. Vrijednosti ovih varijabli prikazuje objekt „ob_text“. „Hpp“ predstavlja zdravlje igrača. Kada je varijabla veća od 1, igrač je živ, a

kada je manja, igrač umire. To, u svakom „step event“-u, provjerava objekt „ob_player“. Vrijednost „hpp“-a smanjuje se svakim pogotkom neprijateljskih „metaka“ ako funkcija u „step event“-u „instance_place“ s objektom „ob_player_ghost“ unesenim kao parametrom objekt „ob_enem_bullet“ (3.2.7.) vrati pozitivan rezultat. Također i ukoliko nekom od neprijateljskih vozila (3.2.7.2.) „instance_place“ s parametrom objekta igrača vrati pozitivan rezultat (ukoliko vozilo pregazi igrača) te „instance_place“ objekta mine (ako igrač stane na minu) (3.2.9.). Početna vrijednost „hpp“ jednaka je 100(%). „Armo“ predstavlja vrijednost pancirke koja gubi vrijednost u igri isto kao i „hpp“, ali ne i zbog automobila. Dok je vrijednost „armo“ veća od nule za svaku izgubljenu vrijednost, usporava pad vrijednosti „hpp“ za duplo manje. Kada igrača pogodi neprijateljski metak, vrijednost „armo“ smanjuje se za 10, a „hpp“ za 5, no ako je „armo“ jednak 0, „hpp“ se smanjuje za 10. Zadana vrijednost „armo“ jednaka je 100(%). „Scor“ varijabla početne je vrijednosti 0. Ona predstavlja „bodove“ koje igrač skuplja kroz igru. Za ubijenog neprijatelja, varijabli „scor“ vrijednost raste za 200, za raznesenu bačvu benzina za 25, za skinutu neprijateljsku zastavu 1000, a za ozljeđivanje civila smanjuje se za 1000.

3.2.3.1. Create event

„Create event“ započinje definiranjem triju varijabli „akt“, „cro“ i „scor“. Tim trima varijablama dodjeljuje se vrijednost 0. Slijedi provjera varijable „global.heltdalje“. Ako je njena vrijednost različita od -50, dodjeljuje se varijabli „hpp“, a „global.armodalje“ varijabli „armo“ i „global.scordalje“ „scor“. Ukoliko je jednaka -50, kao što je objašnjeno u prošlom odlomku, znači da je postavljena na zadanu vrijednost te se varijablama „hpp“, „armo“ i „scor“ dodjeljuju početne vrijednosti (hpp=100, armo=100, scor=0).

3.2.3.2. Step event

Prva radnja „step event“-a je izjednačavanje x i y koordinata „ob_player_ghost“-a i „ob_player“-a. Provodi se bez uvjeta. Kao što je ranije pojašnjeno, igrač upravlja objektom „ob_player“, a „ob_player_ghost“ prema njemu prilagođava sprite-ove koje igrač vidi te zbog toga „ob_player_ghost“ bezuvjetno svaki trenutak igre provodi prateći objekt „ob_player“. Prvo, ovaj „step event“ provjerava drži li igrač tipku „space“ te „hspeed“ objekta „ob_player“. Ukoliko igrač drži tipku „space“ i „hspeed“ je jednak nuli, sprite se mijenja u „sp_player_ghost_triger“ te se „image_speed“, brzina

sprite-a, postavlja na 0,2. „Sp_player_ghost_triger“ je sprite s više slika koje simuliraju animaciju. On prikazuje kako igrač okida okidač puške te ona trza nazad. Razlog provjere brzine je činjenica da se koristi drugi sprite ukoliko igrač hoda. Sljedeće provjerava je li igrač pustio tipku „space“ i „hspeed“ objekta „ob_player“. Ako je rezultat te provjere 'true' i '0', sprite se postavlja na defualt sprite „sp_player_ghost“. Treća provjera provjerava varijablu „opto5“ objekta „ob_player“ (3.2.2.). Ako je vrijednost „opto5“ jednaka 1, provjerava se tipka „c“, a ukoliko je 2, tipka „s“. Ukoliko je jedna od te dvije tipke pritisnuta, „ob_player“ klekne, a „ob_player_ghost“ promijeni sprite u „sp_player_ghost_crouch“ (slika: 29) koji predstavlja igrača kako kleči. „Hspeed“ objekta „ob_player“ se postavlja na nulu kako bi se spriječilo igrača da se kreće dok ne stoji i na kraju varijabli „cro“ dodjeljuje se vrijednost 1. Varijabla „cro“ kasnije u „step event“-u u nekim situacijama odlučuje hoće li igrač, kad se zaustavi, stajati ili klečati. Ovisno o „opto5“ provjeri, provjerava se je li otpuštena tipka „c“ ili „s“. Ako jest, igrač ustaje tako što se vrati početni sprite. „Cro“ dobija vrijednost 1. Sljedeća provjera je vrijednost „opto2“ varijable objekta „ob_player“. O njoj ovisi kojom tipkom se „ob_player“ kreće u lijevo. U ovom slučaju pritiskom na neku od tih tipki, vrijednost „xscale“ varijable mijenja se u -1 što znači da se „ob_player_ghost“ okrene u lijevu stranu bez obzira na trenutni sprite. Sljedeća provjera je „opto3“ varijable i tipki za hodanje u desno. Proces je isti kao i kod prošle provjere, no ovdje se vrijednost „xscale“ varijable postavlja na 1. Sljedeće četiri provjere provjeravaju treba li „ob_player_ghost“ hodati, dakle treba li se aktivirati animacija hodanja. Prve provjere su „opto2“ i „opto3“. „Opto2“ provjerava hodanje u lijevo, a „opto3“ hodanje u desno. Sljedeće dvije provjere ovise o tome rezultatu. Provjeravaju se dvije od tipki „a“, „d“, „left“, ili „right“ i provjerava se horizontalna brzina („hspeed“) „ob_player“-a. Provjera za hodanje desno provjerava ili „d“ ili „right“ tipku i provjerava je li brzina „ob_player“-a veća od 0,1. Ukoliko ta provjera da pozitivan rezultat, aktivira se sprite „sp_player_ghost_walk“ i brzina tog sprite-a postavi se na 0,25. Za hodanje u lijevo provjerava se tipku ili „a“ ili „left“ i provjerava se je li brzina manja od -0,1. Brzina manja od -0,1 znači veća od 0,1 u lijevo, prema negativnim koordinatama. Zadnja provjera „step event“-a provjerava „opto2“ i „opto3“ varijable te, ovisno o njima, tipke kao i iz prošle provjere, no ovaj puta provjerava jesu li otpuštene i kleči li igrač (ranije opisana varijabla „cro“). Ako je neka od tih tipki puštena i igrač ne kleči već stoji, akitivira se defualt sprite „sp_player_ghost“. Ukoliko

kleči u sljedećem ponavljanju „step event“-a, odgovarajuća promjena će aktivirati odgovarajući sprite.

Slika 29: sp_player_ghost_crouch

3.2.4. ob_text

Objekt „Ob_text“ je jedan od važnijih objekata u igri. Kreira ga „ob_player“ u „room start“ event-u (3.2.2.). Služi za prikazivanje važnih podataka. Prikazuje se razina zdravlja i pancir prsluka izražena u postotku, količina prikupljenih bodova, naziv trenutne sobe u kojoj se igrač nalazi i tri sprite-a. Tri sprite-a koja „ob_text“ prikazuje su: „sp_bar“, „sp_sun“ i „sp_metci“. „Ob_text“ sadrži svega dva event-a: „create event“ i „draw“ event.

U **„create event“**-u definira se varijabla „mecigore“ kojoj se pridružuje vrijednost nula.

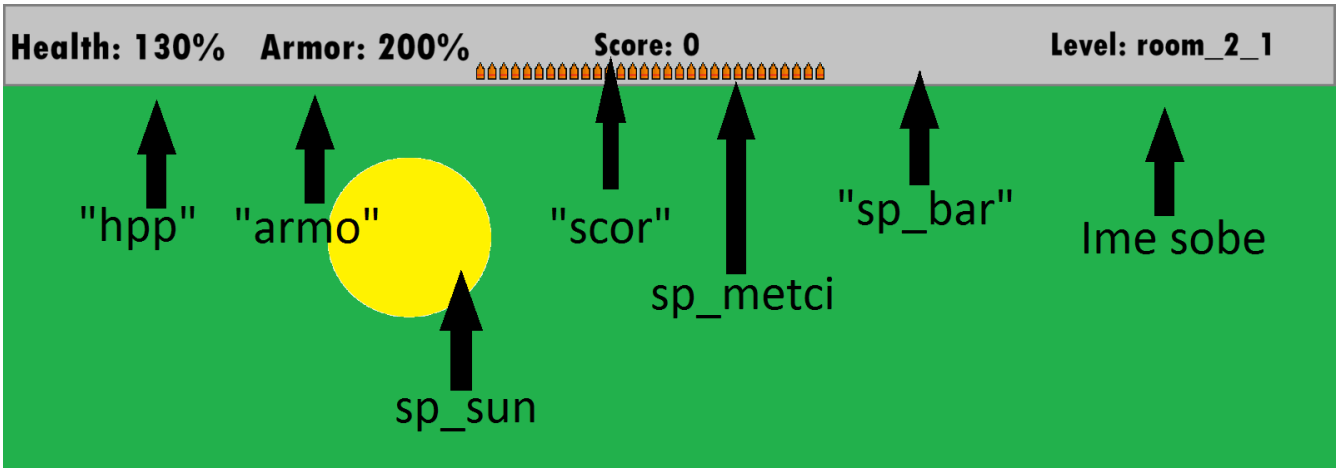
„Draw event“ započinje provjerom postoji li „ob_player“. Tom provjerom postiže se da, ukoliko objekt igrača prestane postojati, nestane i prikaz njegovih podataka. Do te situacije dolazi ako igrač odluči izaći iz igre i vratiti se u glavni izbornik u kojem nema „ob_player“-a bez kojeg nema potrebe za prikazom podataka koje „ob_text“ prikazuje. Prva funkcija koja se pokreće je „draw sprite“ u koju je, kao sprite za prikazivanje, unesen „sp_metci“. On prikazuje koje prikazuje preostale metke u magazinu. Vrijednosti u funkciji su i koordinate na koje se crta sprite. U ovom objektu sve koordinate ovise o aktivnom view-u „0“ pa se tako na primjer, „x“ varijabli dodjeljuje vrijednost „view_xview[0]+ (razlika u pozicijama tekstova ili sprite-ova)“. U slučaju „sp_metci“ „y“ poziciji tog sprite-a dodaje se vrijednost 90, a „x“ poziciji 730. To pozicionira sprite na sredinu ekrana, 90 piksela od vrha. „X“ poziciji oduzima se vrijednost varijable „mecigore“. Za svaki ispaljeni metak iz magazina opisanog kod objekta „ob_player“, vrijednost „mecigore“ raste za 18 te tako pomiče sprite za 18

piksela u lijevo (18 piksela jednako je visini svakog posebnog metka), stvarajući efekt nestajanja zadnjeg metka. Sljedeći sprite, koji generira ovaj event, je „sp_bar“. On služi kao siva pozadina na vrhu ekrana preko koje su ispisani podatci korisni za igru. „Sp_bar“ stvara se u gornjem lijevom kutu „view“-a „0“ na koordinatama (0,0) u odnosu na view. Sljedeće event provjerava varijablu „global.sunny“. Tu varijablu svaka soba definira u „creation code“-u i postavlja joj vrijednost na 0 ili 1. Te vrijednosti unaprijed su ručno postavljene ovisno o tome je li soba u interijeru ili eksterijeru. Ukoliko je vrijednost te varijable jednaka 1, radi se o eksterijeru te se pokreće još jedna u nizu „draw_sprite“ funkcija. Ova funkcija prikazuje sprite „sp_sun“ koji predstavlja sunce na nebu. Nakon toga prikazuju se isključivo tekstovi, no prije toga kao aktivni font definira se „font6“. Sljedeći redovi koda provjeravaju uvjete prikazivanja varijabla „hpp“ i „armo“ objekta „ob_player_ghost“ (3.2.3.). Za svaku posebno, provjerava se je li vrijednost tih varijabli veća od 30. Ako je veća, kao boja teksta definira se crna, no ako je manja od 30 definira se crvena boja. (slika: 30) Nakon definiranja boje u oba slučaju pokreće se „draw_text“ funkcija u koju se kao argument teksta, između ostalog, koristi i jedna od tih varijabli. Razlog mijenjanja boje u crvenu je naglašavanje da je „ob_player_ghostu“ oslabljeno zdravlje ili da mu je oslabio pancir prsluk. Iz primjera ispod, koji prikazuje kod za ispis zdravlja igrača („health“-a), vidi se provjera vrijednosti „hpp“ varijable i ovisnost boje teksta o njoj. Nakon toga, u funkciju „draw_text“ unose se željene koordinate koje su postavljene u odnos s „view“-om „0“. Zadnji argument je string koji ispisuje riječ „Health“ i znak „%“ između kojih se prikazuje promjenjiva vrijednost „hpp“.

```
if(ob_player_ghost.hpp<=30){draw_set_color(c_red);}
else {draw_set_color(c_black);}
draw_text(view_xview[0]+10,view_yview[0]+30,"Health: "+string(ob_player_ghost.hpp)+"%");
```

Slika 30: Primjer koda.

Nakon toga „font5“ se postavlja kao font ispisa. Pokreću se dvije „draw_text“ funkcije. Jedna ispisuje broj sakupljenih bodova („scor“ 3.2.3.), a druge ispisuje naziv sobe u kojoj se igrač nalazi. Razlika fontova 5 i 6 je samo u njhovoj veličini: „font5“ veličine je 40, a „font6“ 45. Oba fonta su „Tw Cen MT condensed“. Slika (31) prikazuje izgled objekta „ob_text“.



Slika 31: Ispisi objekta "ob_text", na njihovim pozicijama na ekranu

3.2.5. Ostali objekti vezani uz igrača

Ako „Ob_player“ promjeni global varijabli „alive“ vrijednost u nula, to jest kad „umre“, stvara objekt „ob_ded“. **„Ob_ded“** ima jednostavnu ulogu bojanja pozadine u crveno u slučaju kad igrač izgubi sve bodove zdravlja kako bi se bolje naglasio kraj igre. „Ob_ded“ na dnu ekrana stvara objekt „ob_player“ u „step event“ provjeri varijable „global.alive“. Sadrži crvenkasti jednobojni sprite („sp_ded“, slika: 32) i samo jedan event. U tom „create event“-u izmjenjuju se vrijednosti triju programskih varijabli. „Xscale“ i „yscale“ se izmjenjuju s defualt vrijednosti 1 u 35 kako bi se sprite razvukao preko cijelog ekrana. Varijabli „vspeed“ vrijednost se postavlja na -3 kako bi se postigao efekt bojanja pozadine s dna ekrana prema gore.

Slika 32: ob_ded

„Ob_magazin“ je objekt koji „ob_player“ stvara u alarm event-u (3.2.2.) kad varijabla istog objekta „magazin“ dosegne vrijednost 0. Predstavlja prazni spremnik municije koji pada na pod nakon što ga „ob_player“ zamjeni za puni. (slika: 33) U „create event“-u postavlja „vspeed“ na 9 što simulira pad. „Step event“ vrši dvije provjere. Prva provjera je provjera pada poput one „ob_player“-a (3.2.2.) koja u svakom

trenutku igre provjerava solid objekte na svome putu prema dolje. Ukoliko provjera pada da pozitivan rezultat, brzinu postavlja na 0, a kut slike okreće za 90 stupnjeva u desno kako bi spremnik horizontalno pao na solid objekt, to jest „pod“, umjesto da ostane u uspravnoj poziciji. Druga provjera provjerava je li udaljenost „ob_player“-a veća od 1850 piksela. Ako jest, objekt se samouništava kako bi se izbjegla prevelika količina objekata u aktivnoj sobi. 1850 piksela udaljenosti je malo više od pogleda „view“-a „0“, dakle „magazin“ se uništava kad je van pogleda igrača plus nekoliko piksela preko toga.

Slika 33: ob_magazin

„Ob_shell“ nastaje prilikom pucanja. Predstavlja praznu čahuru koju puška izbacuje prilikom izbacivanja „metka“. (slika: 34) Stvara je „ob_player“ u alarm event-u pucanja. Kao i „ob_magazin“-u, prilikom nastajanja dodjeljuje joj se vertikalna brzina prema dolje pomoću koje pada, a „Step event“ provjerava solid objekte ispod čahure za vrijeme pada. Ukoliko se ispod čahure nalazi solid objekt, vertikalnoj brzini dodjeljuje se vrijednost 0. Kao i kod „ob_magazin“-a, „step event“ provjerava udaljenost od „ob_player“-a koja, ukoliko je veća od 1850 piksela, pokreće samouništenje čahure na podu.

Slika 34: ob_shell

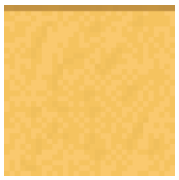
„Ob_bullet“ stvara se istovremeno s „ob_shell“, dakle stvara ga „ob_player“ u „alarm event“-u koji pokreće igrač pritiskom na tipku „space“. Taj objekt predstavlja metak (slika: 35) koji igrač „ispaljuje“ u igrici kako bi „ubio“ neprijatelje (neprijatelji također stvaraju instance metaka, no tu se radi o drugom objektu (3.2.7.)), uništio neprijateljska vozila, aktivirao mine... U „create event“-u provjerava se strana na koju je okrenut „ob_player“ te se metku pridružuje horizontalna brzina 50, odnosno -50 ovisno o rezultatu te provjere. „Step event“ vrši dvije provjere. Prva provjerava objekte na putanji metka na udaljenosti jednakoj njegovoj brzini te ako dobije pozitivan rezultat pokreće funkciju „move to contact“ koja poziciju metka podešava kako bi metak „dotakao“ taj objekt i time aktivirao „instance_place“ funkciju koja se

nalazi u sljedećoj provjeri. Sljedeća provjera provjerava je li „ob_bullet“ „pogodio“ neku važnu metu. Ta provjera sadrži trinaest varijabli koje sadrže vrijednosti „instance_place“ funkcija za različite objekte na koje „ob_bullet“ može utjecati. „Instance_place“ funkcija ne sadrži nikakvu vrijednost dok se na zadanoj poziciji danog objekta ne pojavi objekt koji izvršava tu funkciju. Dakle, ukoliko objekt metka dotakne jedan od objekata pokriven jednom od „instance_place“ funkcija koje „ob_bullet“ sadrži, ta funkcija vraća vrijednost 1 umjesto zadane vrijednosti „noone“. Nakon definiranja tih varijabli i funkcija, „step event“ provjerava promjene vrijednosti s „noone“ na 1 za svaku od tih varijabli. Ako neka od provjera vrati vrijednost jedan, metak je „pogodio“ metu te joj mijenja neku od varijabli koja aktivira reakciju koju pokreće pogođeni objekt. Na primjer: ako metak pogodi neprijatelja, smanjiti će mu „zdravlje“ (3.2.7.) te ako metak pogodi bačvu benzina promjeniti će joj vrijednost varijable „expl“ na jedan što će kod bačve aktivirati eksploziju. (3.2.9.) Ti primjeri detaljnije su objašnjeni u odlomcima tih pojedinačnih objekata. „Ob_bullet“ sadrži još tri event-a: „outside room“, „outside view 0“ i „collision event“ s objektom „ob_sand_parent“. Svaki od tih event-ova pokreće istu funkciju, „instance_destroy()“. Dakle, metak se samouništava ukoliko je van pogleda igrača, van sobe ili ukoliko pogodi „ob_sand_parent“ koji je „parent“ objekt „objekata terena“ (3.2.6.).

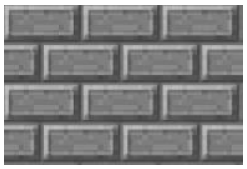
Slika 35: ob_bullet

3.2.6. Objekti terena

„Ob_sand_parent“ je objekt koji se niti jednom ne pojavljuje u igri, već služi kao „parent“ objekt. Njegova jedina uloga je reagiranje s objektom „ob_bullet“ koji uništava. Razlog stvaranja „parent“ objekta je smanjenje količine provjera i kodova u „step event“-u „ob_bullet“-a. „Parent“ je 11 različitih objekata, a neki od njih su: **„ob_sand“, „ob_sand2“, „ob_sand3“, „ob_sand4“, „ob_stone“, „ob_stone2“, „ob_stone3“**. (Slike: 36 i 37)



Slika 36: ob_sand (pjesak)



Slika 37: ob_stone (kameni pod)

Tih 7 navedenih objekata u ulozi su tla, to jest poda po kojem „ob_player“ „hoda“. Jedina karakteristika osim sprite-a im je oznaka „solid“ koju „ob_player“ u „step event“-u provjerava za kolizije. „Ob_sand“ je pješčana podloga u sobama na otvorenom. „Ob_stone“ predstavlja zidove i podove prostorija i zgrada koje se pojavljuju u igri. Sufiksi u nazivima tih objekata označavaju objekte različitih dimenzija. **„Ob_stone_exp“** je također „dijete“ „ob_sand_parent“-a. Označen je kao solid te ima istu teksturu kao i ostali „ob_stone“ objekti. Razlikuje se po tome što u „create event“-u deklarira varijablu „rushi“ i dodjeljuje joj vrijednost 0, a u „step event“-u provjerava promjene vrijednosti te varijable te ukoliko dođe do promjene, objekt se uništava. Objekt koji može promijeniti vrijednost te varijable je objekt vertikalne mine koja „stoji“ na tom zidu i predstavlja eksplozivnu napravu zaljepljenu na zid. **„Ob_box“** i **„ob_boxv“** su dva objekta koji su također „djeca“ „ob_sand_parent“-a. Predstavljaju drvene kutije koje u igri mogu služiti kao zapreka igraču ili pak zaklon (slika: 38).

Slika 38: ob_box

Oba su solid objekti. Jedina razlika im je strana na koju su okrenuti: „ob_box“ dodiruje duljom stranom pod, a „ob_boxv“ stoji vertikalno. Zadnji objekt „dijete“ „ob_sand_parent“-a je **„ob_elevator“** koji se u igri pojavljuje samo jednom, u sobi „room_3_2“. Taj objekt predstavlja teretno dizalo koje igrač aktivira tako što stane na njega i dizalo se počne spuštati dolje. Objekt ima jedan „step event“ koji vrši tri provjere. Prva provjera sadrži „instance_place“ funkciju koja provjerava nalazi li se „ob_player“ poviše objekta dizala. Ako se nalazi, dizalo se krene spuštati dolje brzinom „vspeed=1“. Dizalo se prestaje spuštati ako neka od druge dvije provjere da pozitivan rezultat. Prva od njih također sadrži „instance_place“ funkciju koja provjerava NE nalazi li se „ob_player“ iznad dizala na visini do 112 piksela iznad dizala, a druga provjerava koliziju dizala s objektom za provjeru okreta. „Ob_okret“ jedino služi svrsi zaustavljanja dizala te je igraču nevidljiv te se u ovom slučaju nalazi na istoj visini kao i kameni pod ispod dizala. Zadnji u nizu objekata svrstanih pod objekte terena je **„ob_stone_3iza“**. Jedini je objekt u ovom potpoglavlju koji nije niti „dijete“ „ob_sand_parent“-a niti solid objekt. „Ob_stone_3iza“ nema niti jedan event, već služi samo kao dekoracija. Sprite mu se razlikuje od sprite-a „ob_stone3“ samo po boji: „ob_stone_3iza“ svjetlije je boje te kao takav predstavlja „zid“ koji se nalazi u pozadini (slika: 39).

Slika 39: ob_stone_3iza

3.2.7. Objekti neprijatelja

Objekti „neprijatelja“ su objekti koji se protežu kroz cijelu igru. Glavna su zapreka igraču dok prolazi kroz razine. Kad se igrač približi neprijateljima, oni ga počnu napadati, to jest počnu pucati prema njemu. Dijele se u dvije vrste. Prva vrsta, koja se i najčešće pojavljuje, su pješaci. Pješaci, osim što pucaju na igrača, mogu ga pratiti. Druga vrsta neprijatelja su vozila koja se pojavljuju pred završetak razina. Vozila predstavljaju veću opasnost za igrača. Osim što pucaju na igrača, mogu ga pregaziti, a i teže ih je uništiti nego pješake.

3.2.7.1. Osnovni objekti neprijatelja, pješaci

Objekti: **„ob_isis“**, **„ob_isis_ispod“**, **„ob_isis_iznad“**, **„ob_isis_stoji“**, **„ob_isis_stoji_ispod“** i **„ob_isis_lift_ispod“** predstavljaju najčešći tip neprijatelja. Na prvi pogled izgledaju jednako jer dijele isti sprite (slika: 40) i napadaju igrača ukoliko im se približi. Razlikuju se po tome na koliku udaljenost im igrač treba prići kako bi ga napali te po tome što neki neprijatelji hodaju, to jest kreću se prema igraču, a neki stoje na svome mjestu.

Slika 40: sp_isis

Create event definira varijable „hp“ i „isis_magazin“. „Hp“ varijabli dodjeljuje se vrijednost 5. „Hp“ predstavlja zdravlje objekta neprijatelja koje se smanjuje za 1 svaki put kada ga pogodi metak igrača („ob_bullet“). „Isis_magazin“-u dodjeljuje se vrijednost 20 koja se provjerava i smanjuje u „alarm[1]“ event-u.

Step event postavlja „vspeed“ na „vspeed=vspeed+1.1“ i tako ubrazava objekt prema dolje simulirajući gravitaciju. Zatim se otvara execute code. On prvo provjerava „instance_place“ s objektom „ob_force“ (3.2.9.) koji nastaje prilikom detonacije bačve s benzinom. Ako „instance_place“ da pozitivan rezultat, „hp“ se smanjuje za 4. Druga provjera provjerava vrijednost „hp“-a koji, ako je jednak ili manji od nule, uništava objekt funkcijom „instance_destroy()“ i pritom se aktivira „destroy“ event. Sljedeće provjere su ključna razlika objekata neprijatelja, razlikuju se od objekta do objekta. Provjeravaju udaljenost igrača te prema tome pucaju, to jest aktiviraju „alarm[1]“, a neki od objekata se i kreću prema igraču. Zadnje provjere step event-a su provjere kolizije sa „solid“ objektima. One su jednake već opisanim provjerama objekta „ob_player“. (3.2.3.) „Ob_isis“ i „ob_isis_ispod“ kreću se prema igraču. „Ob_isis“ se počne „kretati“ prema njemu ukoliko mu se približi na udaljenost

manju od 1539 piksela. „Ob_isis_ispod“ reagira na istu udaljenost, no jedino ako se igrač nalazi na jednakoj visini („y“ koordinati). „Ob_isis_ispod“ je koristan ukoliko se u nekoj sobi nalaze staze na različitim visinama. „Ob_isis“ igraču se približi na 839 piksela i počne pucati, a ako mu se igrač odmakne na više od 1539 piksela, „ob_isis“ krene hodati za njim i prestane pucati. „Ob_isis_ispod“ igraču se primakne na 339 piksela prije nego se zaustavi, a počne pucati već na 739 piksela udaljenosti. Prestane pucati ukoliko mu se igrač odmakne na više od 739 piksela. „Ob_isis_stoji“, „ob_isis_stoji_ispod“ i „ob_isis_lift_ispod“ nemaju mogućnost kretanja. Reagiraju na igrača jedino pucanjem te se često nalaze iza raznih zapreka i zbog toga niti nema potrebe da se kreću. „Ob_isis_stoji“ počne pucati na igrača kada mu se igrač približi na manje od 639 piksela, a prestaje pucate kada se igrač odmakne na više od 639 piksela. „Ob_isis_stoji_ispod“ djeluje jednako, ali na udaljenosti od 140 piksela te se zbog toga pojavljuje na mjestima gdje igrač mora skočiti sa višeg kata u neposrednu blizinu. „Ob_isis_iznad“ uspoređuje svoju „x“ koordinatu s „x“ koordinatom „ob_player“-a. Ukoliko je razlika njih manja od 75 piksela, varijabli „hspeed“ dodjeljuje vrijednost 5. Taj objekt nalazi se isključivo na rubovima zidova te nakon što skoči sa zida, provjerava vertikalnu koliziju. U trenutku kad dotakne tlo, počne pucati prema igraču, a „hspeed“ promjeni u zadanu vrijednost, 0. Instance objekta „ob_isis_lift_ispod“ pojavljuju se u samo jednoj sobi, „room_3_2“ . Osnova te sobe je teretno dizalo „ob_elevator“ kojim se igrač spušta dolje kroz tu sobu. „Ob_isis_lift_ispod“ uspoređuje svoju „y“ koordinatu s „y“ koordinatom „ob_player“-a te ukoliko je razlika između te dvije varijable manja od 110 piksela, aktivira alarm[1], to jest počinje pucati. „Ob_isis_lift_ispod“ nema potrebu za provjeravanjem općenite udaljenosti od igrača niti za uspoređivanjem „x“ koordinata zato što se njegove instance jedino nalaze u uskoj sobi gdje igraču mogu prići samo s jedne strane.

Alarm[1] event, koji aktivira „step event“ kad se igrač približi neprijatelju, služi za „pucanje“, to jest stvaranje objekata metaka. Prilikom aktivacije alarma smanjuje se vrijednost varijable „isis_magazin“ za jedan, a zatim slijedi provjera te varijable. Ako je veća od 0, alarm se postavlja na 15, a ako je „isis_magazin“ jednak 0, vrijednost mu se postavlja na početnu vrijednost, 20, alarm se postavlja na 95 i kreira se „ob_magazin_ak“ što predstavlja zamjenu spremnika municije, slično kao i kod „ob_player“-a. Nakon što se kreira „ob_magazin_ak“, taj objekt ima jednake karakteristike kao i ranije opisani „ob_magazin“ (3.2.5.). Jedina razlika su im sprite-

ovi koji predstavljaju spremnike dva različita oružja. Zatim se provjerava „opto1“ varijabla objekta „ob_player“, koji ako vrati vrijednost jedan, aktivira se zvuk „sd_bang“. Posljednja provjera ovog eventa provjerava na koju stranu je objekt okrenut, to jest provjerava se varijabla „image_xscale“. Ako je vrijednost te varijable jednaka 1, to znači da je objekt okrenut na zadanu stranu što je u ovom slučaju nalijevo. Druga mogućnost je -1, to jest objekt je okrenut na desnu stranu. Ovisno o rezultatu „image_xscale“-a, stvara se ili objekt „ob_enem_bullet“ ili „ob_enem_bullet_desno“. Razlika ta dva objekta je u smjeru u kojem idu. Na kraju se stvara objekt „ob_shell“ (3.2.5.) koji predstavlja čahuru koju oružje izbacuje. Pozicija na kojoj se stvara ovisi o „opto1“ te bez obzira na koju stranu je objekt neprijatelja okrenut, čahura se stvara iznad sredine cijevi oružja neprijatelja.

Destroy event aktivira se nakon što se objekt neprijatelja uništi kad varijabla „hp“ dostigne vrijednost 0. Stvara objekte „ob_isis_ded“ i „ob_points_200“ (3.2.5.). Objekt „ob_isis_ded“ sadrži samo jedan „step event“ koji provjerava vertikalnu koliziju sa „solid“ objektima, a predstavlja „ubijenog“ neprijatelja (slika 41).

Slika 41: ob_isis_ded

3.2.7.2. Napredni objekti neprijatelja, vozila

„Ob_himlux“, **„ob_himlux2“** i **„ob_finall“** su objekti koji predstavljaju neprijateljska vozila. Sva tri objekta pojavljuju se jedanput. „Ob_himlux“ je zadnja prepreka u prvoj razini igre, „ob_himlux2“ je posljednja prepreka druge razine, a „ob_finall“ je zadnja prepreka posljednje razine. Razlikuju se po sprite-u te po „hp“-u koji se definira u „create event“-u. „Ob_himlux“ „hp“-u dodjeljuje vrijednost 200, a „ob_himlux2“ i „ob_finall“ 250. Sljedeće karakteristike su im jednake. „Ob_finall“ se razlikuje i po tome što ne reagira na „ob_bullet“ to jest igračeve metke, već isključivo na eksplozije bačvi benzina.

U **„create event“**-u, osim „hp“-a koji predstavlja „zdravlje“ vozila, definiraju se i „hspeed=0“, „alv=1“, „ex=80“, „konfeti=1“ i „image_speed=0.1“. Kreira se objekt „ob_boss_text“ koji u „draw event“-u prikazuje životne bodove, to jest „hp“ neprijateljskog vozila, koristeći fontove „font0“ u crvenoj boji i „font1“ u crnoj.

„Ob_boss_text“ u step eventu provjerava „hp“ vozila, te se uništava kad „hp“ dostigne nulu.

„Step event“ počinje provjerom „instance_place“ s objektom „ob_okret“ koja ukoliko da pozitivan rezultat, „hspeed“ postavlja na zadanu vrijednost, 0. Sljedeća provjera je također „instance_place“, no ovaj puta sa objektom „ob_force“ koji nastaje prilikom detonacije bačve s benzinom. Ako taj „instance_place“ da pozitivan rezultat, „hp“ vozila smanjuje se za 30. Zatim se provjerava „alv“ varijabla koja ima zadanu vrijednost 1. Ta vrijednost postaje jendaka nuli ako vozilo izgubi svo „zdravlje“. Sljedeća provjera provjerava alarm[4]. Ukoliko alarm[4] nije aktiviran, provjerava se udaljenost „ob_player“-a od vozila. Ako je udaljenost manja od 1000 piksela ili 1300 u slučaju „ob_himlux2“, kreira se brojač „ddd“ kojem se dodjeljuje vrijednost 1. Ta vrijednost se zatim provjerava te se alarmi „[1]“ i „[4]“ postavljaju na 1 što ih aktivira sljedeći trenutak. „Ddd“-u dodjeljuje se vrijednost nula kako se ne bi alarmima i sljedećem „step event“-u dodijelila vrijednost 1.

Nakon ovih provjera slijedi provjera „hp“ varijable koju se provjerava je li dostigla vrijednost 0. Ukoliko jest, to znači da je vozilo „uništeno“. Sprite vozila mijenja se u „sp_himlux_ded“ ili „sp_finall_ded“, sprite koji predstavlja uništeno i izgoreno vozilo. Varijabli „alv“ vrijednost se postaje 0 kako bi se sprječilo vozilo da nastavi pucati prema igraču. „Hspeed“ također se postavlja na 0 kako bi se uništeno vozilo zaustavilo. Vrijednost varijable „ex“ smanjuje se za 1. Varijabla „konfeti“ koja je deklarirana u „create event“-u u ovom slučaju služi kao brojač. Provjerava joj se početna vrijednost koja se nakon izvršavanja zadane funkcije mijenja kako se ne bi ponavljala. „Slika 42“ prikazuje kod stvaranja „animacije“ koja se aktivira nakon što se vozilo „uništi“. Ako je „konfeti“ jednak 1, pokreće se „instance_create“ funkcija koja kreira objekt „ob_isis_garica“. On predstavlja neprijateljskog vojnika kako izliječe iz vozila pogurnut eksplozijom vozila. Zatim se provjerava varijabla „ex“ kojoj svako ponavljanje „step event“-a smanjuje vrijednost za jedan. Provjerava se je li „ex“ veći od 0 i je li vrijednost „ex“ djeljiva s brojem 10. Ako je na prednjem dijelu vozila (iznad motora), stvara se eksplozija, „ob_explosion“. Djeljivost s 10 se provjerava kako bi se eksplozije pojavile svakih 10 trenutaka, a ne odmah jedna iza druge. Ukoliko je vrijednost „ex“ jednaka 50, stvara se i jedna eksplozija na stražnjem dijelu vozila. Provjerava se i „opto1“ koji, ako vrati vrijednost 1, aktivira zvuk „sd_explosion“.

```
if(konfeti==1) { instance_create(x+50,y-100,ob_isis_garica); konfeti=0;}
if(ex>0 and ex%10==0){
    if(ex==50){instance_create(x+160,y-140,ob_explosion); }
    instance_create(x-240,y-140,ob_explosion);
    if(ob_player.opto1==1){
    audio_play_sound(sd_explosion, 10, false);}}
```

Slika 42: primjer koda, nastanak eksplozija

Igrač riskira da ga pregazi vozilo ukoliko mu se previše približi. Objekt vozila, ako je varijabla „alv“ jednaka jedan, provjerava rezultat „instance_place“ funkcije koja provjerava nalazi li se u istom prostoru kao i igrač „ob_player“. Ukoliko ta funkcija da pozitivan rezultat, svaki „step event“ igrač gubi 3 boda zdravlja, dakle varijabla „hpp“ objekta „ob_player_ghost“ smanjuje se za tri, sve dok se igrač ne izmakne vozilu. Zadnja provjera ovog „step event“-a provjerava je li vozilo uništeno, to jest provjerava je li vrijednost varijable „alv“ postala 0. Ukoliko jest, zvuk motora „sd_car“ koji pokreće „alarm[1]“ se gasi.

Alarm[1] nakon što je aktiviran, ako su vrijednosti varijabli „alv“ i „opto1“ jednake 1, pokreće zvuk „sd_car“ koji simulira zvuk rada motora. Također postavlja svoju vrijednost na 390 što je jednako trajanju zvuka „sd_car“. Ukoliko udovolji sve uvjete zvuk pokreće ponovo.

Alarm[4] nakon aktivacije provjerava „alv“. Ako je vrijednost „alv“ jednaka 1, pokreće „instance_create“ funkciju koja kreira „metak“ („ob_enem_bullet“ (3.2.7.) u slučaju „ob_himlux“, a „ob_himlux2_bullet“ u slučaju „ob_himlux2“ ili „ob_finall“) koji se ispaljuje u smjeru igrača. „Ob_himlux2_bullet“ se razlikuje od „ob_enem_bullet“ po tome što primjećuje objekte igračevih saveznika koji se nalaze u specifičnoj situaciji gdje i „himlux2“. Alarm si zatim podešava vrijednost na 11 i provjerava „opto1“ varijablu koja ukoliko vrati vrijednost 1, aktivira zvuk eksplozije „sd_bang“ (3.2.10.).

3.2.8. Dekoracijski objekti

Većina dekoracijskih objekata ne sadrži događaje niti se spominje u funkcijama nekih drugih objekata. Jedina svrha dekoracijskog objekta je kako bi se u igri nalazilo više detalja, dakle kako bi se poboljšao izgled igre. Dekoracijski objekti nemaju nikakvu ulogu u prelaženju igre. Ne pomažu niti ometaju igrača, za razliku od objekata

svrstanih u „ostale“ objekte koji su opisani u sljedećem odlomku. Dekoracijski objekti nisu označeni kao „solid“ objekti.

3.2.9. Ostali objekti

Pod ostale objekte spada skupina međusobno različitih i sličnih objekata koji nisu svrstani ni u jednu prethodnu skupinu. Od dekoracijskih objekata se razlikuju po tome što utječu na igru, bez obzira na to je li im uloga primarno dekoracijska.

Objekti bodova su objekti koje kreiraju razni objekti nakon što ih igrač uništi ili se kreiraju nakon što igrač završi razinu igre. Pojavom nekog od objekta bodova mijenja se vrijednost varijable igračevih bodova „scor“. Postoji pet različitih objekata bodova, četiri povećavaju broj bodova (za 25, 200, 1000 i 5000) i jedan smanjuje (za 1000). Objekti bodova nemaju sprite, već je u „draw event“-u definiran tekst koji prikazuju. Objekti koji donose bodove crnom bojom prikazuju brojku jednaku bodovima koje donose, a objekt koji odnosi 1000 bodova to ispisuje crvenom bojom i s predznakom minus. U „create event“-u „vspeed“ varijabli se pridružuje vrijednost -5, „alarm[1]“ koji nakon aktivacije samouništava objekt se namješta na 25 i igraču se dodjeljuju odgovarajući bodovi.

„Ob_chest“ predstavlja medicinsku kutija kojom si igrač obnavlja zdravlje („hpp“ varijabla objekta „ob_player_ghost“). (3.2.3.) Sadrži „step event“ u kojem se provjerava „instance_place“ funkcija s objektom igrača kao argumentom. Ako „instance_place“ vrati pozitivan rezultat, igraču se zdravlje poveća za 30%, „ob_chest“ se uništi i aktivira se zvuk „sd_health“. (3.2.10.)

„Ob_bullproof“ je objekt sličan „ob_chest“, no predstavlja pancir prsluk. Kao i „ob_chest“, sadrži „step event“ s „instance_place“ provjerom. Ako „instance_place“ vrati pozitivan rezultat, samouništi se i aktivira isti zvuk kao i „ob_chest“. Razlikuje se po tome što „ob_bullproof“ poveća snagu pancir prsluku koji igrač nosi, to jest varijablu „armo“ objekta „ob_player_ghost“ poveća za 100%.

„Ob_force“ nevidljivi je objekt veličine 515x472 piksela, koji stvara eksplozije, a samouništi se za 2 trenutka u igri (~0.033 sec). „Force“ predstavlja silu eksplozije nakon aktivacije mine ili bačve s benzinom koja ranjava neprijateljske objekte.

„Ob_mine“ predstavlja nagaznu minu te je jedna od prepreka igraču i ozlijedi ga ukoliko stane na nju. U „create event“-u definira brzinu kojom se prikazuje animacija.

U „step event“-u „instance_place“ funkcijom provjerava je li igrač „stao“ na minu. Ako je igrač „stao“ na minu kreira se „ob_force“ objekt koji ozljeđuje neprijatelje blizu izvora eksplozije, a igraču prvo provjerava stanje pancir prsluka. Ako igrač nema pancir prsluk, odnosno varijabla „armo“ jednaka je nuli, mina mu oduzima 70% zdravlja („hpp“). Ukoliko je stanje pancir prsluka bolje ili jednako 60% („armo“>=60), igraču se oduzima 60% pancir prsluka i 40% zdravlja. Ako je stanje igračevog pancir prsluka veće od 0 i manje od 60%, oduzme mu se sav pancir prsluk i vrijednost zdravlja za pola vrijednosti pancir prsluka manja od 70. Dakle ako je vrijednost pancir prsluka (p), a zdravlje (z), ukoliko igrač stane na minu vrijedi sljedeća formula:

p=p-p(max=60); z=z-(70-p/2);

Iz formule se vidi da, s obzirom na to da je MAX od (p/2)=30, igrač ukoliko stane na minu u najboljem slučaju gubi 40% zdravlja. Nakon tih provjera mina aktivira zvuk „sd_bang“ (3.2.10.).

„Ob_barrel“ je objekt koji predstavlja „bačvu“ benzina. Ako je pogodi igračev „metak“, ona eksplodira. Te zbog toga može biti korisna ukoliko se objekti neprijatelja nalaze u blizini zato što ih može značajno ozlijediti. U „create event“-u definira varijablu „exp“ i dodjeljuje joj vrijednost 0. Objekt metka koji ispali igrač, „ob_bullet“, provjerava u svom „step event“-u „instance_place“ funkciju za provjeru „ob_barell“-a, koja ako vrati pozitivan rezultat, vrijednost „exp“ mijenja u 1. U „step eventu“ „ob_barrel“ provjerava je li došlo do promjene u vrijednosti „exp“. Ako dođe do promjene kreira se sila eksplozije „ob_force“ i objekt bodova „ob_points_25“ koji povećava igračevu „scor“ varijablu za 25, a sprite „bačve“ se promjeni u tamniju, „izgorenu“ verziju originalnog sprite-a. „Vspeed“ objekta se promjeni u -30, a zakrivljenje sprite-a pomoću „random“ funkcije se promjeni za neku od vrijednosti između -40 i +40 i to predstavlja nestabilno lansiranje. Varijabli „exp“ vrijednost se vraća u 0 kako se prošle radnje ne bi ponavljale. „Ob_barrel“ se samouništi nakon što napusti prostor sobe.

„Ob_explosion“ je vizualna prezentacija eksplozije. Sprite se sastoji od šest slika koje predstavljaju različite stadije eksplozije. U „create event“-u postavlja brzinu sprite-a na 0.5 što znači da se slika promjeni svaki drugi trenutak igre i postavlja „alarm[2]“ na 12, što aktivira alarm nakon 12 trenutaka, odnosno nakon prikaza animacije. Aktivacijom „alarm[2]“ objekt se samouništava.

3.2.10. Zvukovi

U projektu se nalazi sedam različitih zvukova: **„sd_health“**, **„sd_explosion“**, **„sd_bang“**, **„sd_car“**, **„sd_heli“**, **„sd_metal“** i **„sd_wind“**.

„Sd_health“ pozivaju dva objekta „ob_chest“ i „ob_bullproof“. „Sd_helth“ predstavlja zvuk „prikupljanja“ objekta, koji su u ovom slučaju medicinska torba za i pancir prsluk. „Sd_explosion“ je zvuk eksplozije. Aktiviraju ga objekti mine „ob_mine“ i bačvi s benzinom „ob_barrel“. „Sd_bang“ najčešće se može čuti u igri jer predstavlja zvuk eksplozije iz oružja te se čuje svaki puta kad neki od objekata „puca“ na drugi objekt. Aktivira ga igrač, saveznici igrača i njihovi neprijatelji. „Sd_car“ zvuk je automobila koji se u igrici može čuti ukoliko je neko od neprijateljskih vozila u blizini. „Sd_heli“ se može čuti samo na početku i na kraju igre jer je to zvuk motora vojnog helikoptera vojske. „Sd_metal“ je zvuk koji se može čuti kad objekt metka pogodi „željezne“ objekte. Posljednji od zvukova je „sd_wind“ koji predstavlja zvuk vjetra. Može se čuti jedino u razini „3_1“ gdje se pojavljuje vjetar popraćen pješčanom olujom.

4. Zaključak

Platforme za izradu igara poput Game Makera uveliko olakšavaju njihovu izradu. Game Maker se pokazao kao odlična i relativno jednostavna platforma za izradu igara koja nudi mnoge mogućnosti. Osim što olakšava izradu igre, olakšava i učenje te mnogima nudi mogućnost da nauče izrađivati jednostavnije igre. Game Maker Language nudi mnoge mogućnosti koje su detaljno opisane na službenoj stranici Game Maker-a. Prednost korisnicima koji nemaju prethodnog iskustva s programiranjem, a odluče se za GML umjesto „drag and drop“, je ta što GML „oprašta“ situacije poput izostavljene točke i zareza na kraju linije ili što ne traži programe da definiraju vrstu varijable koju deklinira. To je također i nedostatak zbog toga što je prilikom izvršavanja programa potrebno prevoditi sve te „oproste“ te zbog toga izvršavanje programa traje nešto dulje. Najčešće programi koje treba prevoditi iz GML-a ne budu jako veliki pa to i nije veliki nedostatak. Početnici koji se odluče za „drag and drop“ umjesto GML-a puno su ograničenijih mogućnosti, no prikaz im može biti puno jednostavniji i lakši za razumjeti dok ne nauče kodiranje. Također im je koristan kao alat kojim mogu razumjeti programske algoritme i način na koji programi rade.

U ovom radu uvodno su opisane osnovne karakteristike Game Makera i GML-a, njihove prednosti i nedostaci. Zatim je pojašnjeno značenje objekta, sprite-ova, soba i ostalih resursa potrebnih za izradu igre. Objašnjeni su primjeri različitih vrsta funkcija te vrsta i tipova varijabli. Detaljno je opisana i pojašnjena igra te najvažniji objekti igre, njihovi sprite-ove, funkcije, zvukovi i događaji.

Prvi korak u izradi projekta bio je odabrati željeni žanr igre. Prema odabranom žanru potrebno je bilo osmisliti mehaniku i izgled igre. Nakon što se osmisli izgled igre, potrebno je u nekom od programa za obradu slika dizajnirati i nacrtati sprite-ove. Za ovaj projekt korišteni su Microsoftov „Paint“, „Gimp“ i Game Makerov „Sprite Editor“. „Paint“ je uglavnom korišten za crtanje slika, „Gimp“ za dodavanje raznih efekata, a „Sprite Editor“ je primarno korišten za dodavanje više slika za jedan sprite kako bi se postigao efekt animacije. Nakon dizajna na red dolazi mehanika. Prvi korak kreiranja mehanike bio je kreiranje soba i pogleda (eng. view) za te sobe te definiranje pravila soba kao što je brzina sobe i „creation code“. Slijedilo je kreiranje objekata za

početnu sobu glavnog izbornika, a zatim i kreiranje objekata koji se pojavljuju u samoj igri. Većini kreiranih objekata dodjeljuju se događaji i funkcije kako bi mogli obavljati svoju ulogu.

Kao jedan od najzahtjevnijih izazova prilikom kreiranja igre može se navesti traženje i popravljanje „bug“-ova, to jest grešaka u kodu. Greške u kodu pojavljivale su se gotovo svaki puta prilikom kreiranja novih segmenata koda. Neke greške su bile veoma uočljive i jednostavne za ispraviti te nisu predstavljale velike poteškoće, ali u nekoliko slučajeva bilo je grešaka koje su zahtijevala višesatne napore kako bi bile pronađene ili ispravljene.

Jednostavnost i kvaliteta Game Makera može pomoći mnogima da nauče programirati i izrađivati igre te tako utjecati na budućnost tržišta video igara.

Sažetak

Cilj ovog rada bila je izrada igre u okruženju Game Maker.

Uvodni dio rada objašnjava što su to računalne igre i uspoređuje ih na s drugim vrstama video igara na tržištu. U uvodu su također navedene neke prednosti i nedostaci računalnih igara u odnosu na video igre za konzole.

Glavni dio rada opisuje kako koristiti Game Maker, što su objekti, sprite-ovi, sobe i ostali resursi, što je GML i kako ga koristiti. Također su opisane najvažnije funkcije, varijable i način njihovog korištenja. Zatim je objašnjen tijek igre te koji objekti i drugi resursi su korišteni.

Ključne riječi: Game Maker, GML, sprite, objekt, događaj, funkcija, varijabla.

Abstract

Purpose of this bachelors thesis was to create a video game using Game Maker.

In the introduction of this work, it is explained what computer games are and they were compared with other types of videogames. Also, some advantages and disadvantages were stated and compared to console games.

The main part explains how Game Maker works, what are objects, sprites, rooms and other resources, what is GML and how to use it. Most important functions, variables are defined and it is explained how to implement them. Then the course of the game is described as well as what objects and other resources were used.

Keywords: Game Maker, GML, sprite, object, event, function, variable.

Literatura:

Knjige:

1. Ford J.L. Jr. (2010), Course Techology Ptr., „Getting Started With Game Maker“.
2. Gregory J. (2018), CRC Press, „Game Engine Architecture, Third Edition“.
3. Vinciguerra D.C. & Howell A. (2018), CRC Press, „The Game Maker Standard“.

Web izvori:

1. Game Maker uputstva, dostupno na: https://docs.yoyogames.com/source/dadiospice/002_reference/index.html
2. Godot Docs, dostpuno na: https://docs.godotengine.org/en/3.0/classes/class_node.html?highlight=node
3. Lisa A. (2017), GoBankingRates, „Happy National Video Game Day — See the Top Franchises of All Time“, dostpuno na: https://www.gobankingrates.com/making-money/business/highest-grossing-video-game-franchises-microsoft-sony/
4. Tanant F. (2018), WebsiteToolTester, „The best game engines for beginners“, dostpuno na: https://www.websitetooltester.com/en/blog/best-game-engine/
5. The International Arcade Museum (2018), Contra, dostupno na: https://www.arcade-museum.com/game_detail.php?game_id=7389
6. Wijman T. (2017), Newzoo, „New Gamin Boom: Newzoo Ups Its 2017 Global Gamers Market Estimate...“, dostupno na: https://newzoo.com/insights/articles/new-gaming-boom-newzoo-ups-its-2017-global-games-market-estimate-to-116-0bn-growing-to-143-5bn-in-2020/

Preuzeti materijali:

1. Svi zvukovi korišteni u igri preuzeti su s „https://freesound.org/“.
2. Teksture zida pridružene objektima „ob_stone“, „ob_stone2“, „ob_stone3“, „ob_stone3iza“ i „ob_stone_exp“ vlasništvo su Microsofta i sadrže autorska prava.

Popis slika: