VIDZEMES AUGSTSKOLA
INŽENIERZINĀTŅU FAKULTĀTE

PIRMĀS PERSONAS ŠAUŠANAS SPĒLES IZSTRĀDE

GADA PROJEKTS

Autors: Emīls Kuzmans
Studenta apliecības nr: IT19036
Darba vadītājs: Mg. Sc. comp Lauris Taube

VALMIERA 2021

KOPSAVILKUMS

Studiju gaitā ne katram ir viegli apgūt programmēšanu veidojot tradicionālas datorprogrammas. Kā alternatīvu var izmantot spēļu izstrādi, kas studentam var likties patīkamāka un tuvāka, un tādējādi ļaut vieglāk apgūt programmēšanu un motivēt studentu programmēt. Tāpēc autors vēlas apgūt spēles izveides procesu un iegūt zināšanas par spēles dziņiem.

Galvenais darba mērķis ir izstrādāt pirmās personas šaušanas spēles prototipu. Teorētiski izpētīt spēles izveides procesu un pieejamos dziņus spēles izveidei. Darbam tika izmantots Unity dzinis pēc iespējamo dziņu izpētes. Rezultātā ir izstrādāts funkcionējošs spēles prototips, kurš var tikt uzlabots. Tika iegūtas zināšanas par spēles izveides procesu un zināšanas par Unity dziņa izmantošanas iespējām.

Iesniegtais gada projekts satur 36 lapas (bez pielikumiem), 17 attēlus, divas tabulas, četrus pielikumus. Gada projekts izstrādāts: no 2021. gada janvāra līdz jūnijam, Valmierā – Vidzemes Augstskolā.

SUMMARY

During studies, it is not easy everyone to learn programming by creating traditional computer programs. As an alternative, game development can be used, which can be more enjoyable and closer to the student, and thus make it easier to learn programming and motivate the student to program. So, the author wants to learn about the process of creating a game and acquire knowledge of the drivers of the game.

The main goal of the project is to develop a prototype for the first-person shooting game. Research and explore the process of creating a game and the available game engines for creating a game. After the research Unity game engine was choses for creating a game. As a result, a functioning prototype of the game has been developed that can be improved. Knowledge was gained about the game creation process and knowledge about the possibilities of using the Unity game engine.

The submitted annual project contains 36 pages (without attachments), 17 images, two tables, four attachments. The annual project was developed from January to June 2021, in Valmiera – Vidzeme University of applied sciences.

РЕЗЮМЕ

Во время учебы непросто научиться программированию, создавая традиционные компьютерные программы. В качестве альтернативы можно использовать разработку игр, которая может быть более приятной и близкой для учащегося и, таким образом, облегчить изучение программирования и мотивировать учащегося к программированию. Итак, автор хочет узнать о процессе создания игры и получить знания о драйверах игры.

Основная цель проекта - разработать прототип игры-стрелялки от первого лица. Изучите процесс создания игры и доступные игровые движки для создания игры. После исследования для создания игры был выбран игровой движок Unity. В результате был разработан действующий прототип игры, который можно улучшать. Были получены знания о процессе создания игры и о возможностях использования игрового движка Unity.

Представленный годовой проект содержит 36 страниц (без приложений), 17 изображений, две таблицы, четыре приложения. Годовой проект разрабатывался с января по июнь 2021 года в Валмиере - Видземский университет прикладных наук.

SAĪSINĀJUMI UN ATSLĒGVĀRDI

FPS - First person shooter

3D - Trīs dimensiju

2D - Divu dimensiju

Unity - Spēļu dzinis

Unreal Engine – Spēļu dzinis

AI – Mākslīgais intelekts

NPC – Spēlētāji, ko vada dators

Game jam - Konkurss, kurā dalībnieki mēģina izveidot videospēli no nulles

Kickstarter - Radošo projektu finansēšanas platforma

C# - c-sharp (programmēšanas valoda)

VFX - vizuālie efekti

Collider – Unity komponente, kas definē objekta formu fizisko sadursmju nolūkiem.

Character Controller – Unity komponents, kura funkcija ir pārvietot spēlētāju atbilstoši videi, tas neizmanto fiziku.

Materiāls - Unity komponente, kas nosaka, kā izskatīsies 3D objekts.

DeltaTime - Sekunžu skaits, kas nepieciešams, lai Unity dzinis apstrādātu iepriekšējo kadru.

Frame Rate - Frekvence (ātrums), kādā displejā parādās secīgi attēli, ko sauc par kadriem.

Spēles Objekts – Unity pamata objekti, kas kalpo kā konteiners komponentēm un reālajām objekta funkcijām.

Vektors – Lielums, kam ir virziens, izmanto, lai norādītu objekta pozīciju. To var lietot arī virziena noteikšanai un rotācijai.

Transform - Unity komponente, kas piemīt visiem spēles objektiem un satur datus par pozīciju, rotāciju un mērogu.

GitHub – Platforma, kur iespējams ievietot kodu un kuru var izmantot versiju kontrolei, darbu koplietošanai un kopīgu darbu veikšanai.

Unity Layer – Layer unity palīdz spēles objektiem norādīt to funkcionalitāti, piemērām, kuri slāņi spēles objektam ir jāignorē, vai nav redzami kamerā. Ar layer palīdzību iespējams veidot funkcionalitāti, kā sadursmes noteikšana.

UI - lietotāja interfeiss.

SATURS

IEVADS

Spēles izveide no nulles ir sarežģīts process spēļu izstrāde ir tāda, ka tajā ir pārāk daudz zināšanu jomu. Piemēram, programmēšana, fizika, mūzika, modelēšana, māksla, animācija, spēļu dizains, projektu vadība. Un jo sarežģītāka ir spēle, jo vairāk vajadzīgas zināšanas. Iespēja, ka viena persona zina visu, kas nepieciešams spēles izstrādei, ir diezgan maza. (Tony Ortega 2019) Pat visvienkāršākajai spēlei ir nepieciešams vismaz 100 stundu darbs, kas nozīmē, ka jums jāpārdod vairāk nekā 600 eksemplāru (katru pa pieci ASV dolāri). Izklausās reāli, bet grūtības pakāpi šajā pašreizējā situācijā var redzēt nākamajā piemērā. Tas ir, tāpat kā tuvoties svešiniekam uz ielas un mēģināt pārdot grāmatu. Un pat šajā gadījumā jums ir priekšrocība no kāda reāllaika kontakta un konkurences trūkuma konkrētajā brīdī. Ir acīmredzams, ka internetā simtiem cilvēku vienlaikus cīnās par potenciālā pircēja uzmanību. (Alex Twofaced 2019) Tas rada sava veida risku izveidot produktu, ko neviens neizmantos, tāpēc ieteicams izstrādāt vismaz dažas vienkāršākas spēles, kas kalpotu kā mācību process, lai iegūtu pamat zināšanas tālākiem projektiem. Autors vēlas apgūt prasmi un pieredzi spēļu veidošanā, kā arī izprast procesu, kas jāveic sekmīgai spēles izveidei. Studiju gaitā ne katram ir viegli apgūt programmēšanu veidojot tradicionālas datorprogrammas, tās var likties bezjēdzīgas, vai garlaicīgas. Kā alternatīvu var izmantot spēļu izstrādi, kas studentam var likties patīkamāka un tuvāka, un tādējādi ļaut vieglāk apgūt programmēšanu un motivēt studentu programmēt.

Mūsdienās videospēles vēl joprojām tiek uzskatītas par briesmīgu ieradumu un tiek liegtas bērniem, jo mediji liek domāt, ka tās bojā bērnus, bet realitātē situācija ir pavisam savādāka. Videospēlēm ir ļoti daudz plusu un labo ietekmju uz cilvēka smadzenēm un labsajūtu, tās spēj uzlabot sociālās prasmes palīdz novērst garīgo slimību efektus, uzlabot problēmu risināšanas prasmes. Daudzspēlētāju spēles kļūst par virtuālām sociālajām kopienām, kur ātri jāpieņem lēmumi par to, kam uzticēties un kam nē, vienkāršas spēles, kurām ir viegli piekļūt, piemēram, "Angry Birds", var uzlabot spēlētāju noskaņojumu, veicināt relaksāciju un novērst trauksmes sajūtu, spēlējot stratēģiskas videospēles uzlabojās problēmu risināšanas prasme un atzīmes skolā uzlabojās, radošumu arī veicināja jebkura veida videospēļu spēlēšana. (Lisa Bowen 2014) Tāpēc, darba autors vēlas papildus aplūkot datorspēļu labo un slikto ietekmi uz cilvēku un iespējams ievietot kādu no ietekmes aspektiem savā spēlē.

Darba mērķis:

Galvenais darba mērķis ir izstrādāt pirmās personas spēli ar šaušanas elementiem.

Darba uzdevumi:

1. Izpētīt un analizēt nepieciešamo programmatūru.
2. Izstrādāt pirmās personas spēli
3. Veikt spēles testēšanu

Metodes:

1. Teorētisko materiālu izpēte
2. Programmēšana
3. Testēšana.

TEORĒTISKĀ DAĻA

1. SPĒĻU DZINIS

Spēļu dzinis ir programmatūra, kas palīdz spēļu izstrādātājam, veicot bieži sastopamus ar spēlei saistītus uzdevumus, piemēram, spēles fiziku, renderēšanu, lietotāja informācijas ievadi.

Dzinis piedāvā atkārtoti lietojamas spēles sastāvdaļas, kā sadursmju noteikšana starp objektiem, modeļu ielāde. Spēli veido šo sastāvdaļu mijiedarbība un nozīme.(Jeff Ward, 2008)

1.1. UNITY DZINIS

Unity 3D ir platforma spēļu izstrādei. Tā ir ieguvis milzīgu spēļu izstrādātāju kopienas uzmanību ar savām funkcijām, atbalstošu cenu noteikšanu un spēcīgām iespējām videospēļu izveidei. Ar spēju darboties kā starp platformu spēļu izstrādes rīku, tas ietaupa izstrādātāju laiku, kas citādi tiktu tērēts, izstrādājot spēles atsevišķām platformām. (Anurag 2018)

Programmēšanas valodas: Unity atbalsta C#, un vairākas citas.NET valodas var tikt lietotas, ja tās var sastādīt saderīgu DLL. Kā arī Bolt vizuālo skriptu izstrāde.

Attēls 1 Unity Spēļu dziņa lietotāja UI

(Unity Technologies 2021)

A – Rīkjosla nodrošina piekļuvi vissvarīgākajām darba funkcijām. Kreisajā pusē tas satur pamata rīkus skata manipulēšanai. Centrā atrodas atskaņošanas, pauzes un soļu vadība. Labajā pusē esošās pogas ļauj piekļūt Unity Collaborate, Unity Cloud Services un jūsu Unity kontam, kam seko slāņu redzamības izvēlne un visbeidzot Redaktora izkārtojuma izvēlne (kas nodrošina dažus alternatīvus izkārtojumus Redaktora logiem un ļauj saglabāt pašu pielāgotus izkārtojumus).

B – Hierarhijas logs ir hierarhisks teksta attēlojums katram spēles objektam ainā. Katram skatuves vienumam ir ieraksts hierarhijā, tāpēc abi logi ir savstarpēji saistīti. Hierarhija atklāj struktūru, kā spēles objekts pievienojas viens otram.

C – Spēles skats simulē to, kāda izskatīsies jūsu pēdējā renderētā spēle, izmantojot jūsu ainu kameras. Noklikšķinot uz pogas Atskaņot, sākas simulācija.

D – Ainas skats ļauj vizuāli orientēties un rediģēt jūsu ainu. Sižeta skats var parādīt 3D vai 2D perspektīvu atkarībā no projekta veida, pie kura strādājat.

E – Inspektora logs ļauj apskatīt un rediģēt visas pašlaik atlasītā spēles objekta īpašības. Tā kā dažāda veida spēles objektiem ir dažādas rekvizītu kopas, inspektora izkārtojums un saturs loga maiņa katru reizi, kad atlasāt citu spēles objektu.

F – Projekta logs parāda jūsu satura bibliotēku, kas ir pieejama izmantošanai jūsu projektā. Importējot saturu savā projektā, tie tiek parādīti šeit.

G – Statusa josla nodrošina paziņojumus par dažādiem Unity procesiem un ātru piekļuvi saistītajiem rīkiem un iestatījumiem.

(Unity Technologies 2021)

Izmaksas: Unity ir pieejami četri darbības plāni

1. Personal, kas ir par brīvu.
2. Plus $399 gadā katrai personai.
3. Pro $1,800 gadā katrai personai.
4. Enterprise $200 mēnesī katrai personai.

Plāni ir jāmaina, ja ieņēmumi vai finansējums sasniedz noteiktu summu, ja ieņēmumi vai finansējums ir mazāki par $100 tūkstošiem gadā tad var palikt pie Personal plāna, ja ieņēmumi vai finansējums pārsniedz $100 tūkstošus, bet nepārsniedz $200 tūkstoši jāizmanto Plus plāns. Ja ieņēmumi vai finansējums ir lielāku par $200 tūkstošiem jāizmanto Pro vai Enterprise. Enterprice ir domāts lielām komandām minimums desmit cilvēkiem.

Unity Asset Store ir pieaugoša satura objektu bibliotēka. Gan Unity Technologies, gan kopienas locekļi izveido šos satura objektus un publicē tos veikalā. Veikalā ir dažādu veidu satura objekti, sākot no tekstūrām, animācijām un modeļiem līdz visam projekta piemēriem, apmācībām un redaktora paplašinājumiem. Ir pieejams bezmaksas un komerciālā satura objektu kopums, ko var lejupielādēt tieši savā Unity projektā. Kā arī ir iespēja kļūt par izdevēju Asset veikalā un pārdot savus Unity veidotos materiālus.(Unity Technologies 2 2021)

Unity satura objekti ir objekti, kuru varat izmantot savā spēlē vai projektā. Satura objekts var nākt no faila, kas izveidots ārpus Unity, piemēram, 3D modelis, audiofails, attēls vai jebkura cita veida faili, kurus Unity atbalsta. Ir arī daži līdzekļu veidi, kurus varat izveidot Unity, piemēram, Animator Controller, Audio Mixer vai Render Texture. Satura objektu veikals ir sakārtots dažādos pieejamos satura objektu veidos:

- 3D saturs

  3D satura objektu sadaļā ietilpst transportlīdzekļi, tēli, rekvizīti, veģetācija un animācijas. Unity humanoīdā animācijas atkārtota mērķauditorijas atlase nozīmē, ka jūs varat sajaukt un saskaņot dažādu avotu tēlus un animācijas.

- 2D saturs

  2D satura objektu sadaļā ietilpst tekstūras, tēli, vide, fonti, materiāli un lietotāja saskares elementi.

- Papildinājumi

  Papildinājumi ir uzlabotas metodes, kuras varat importēt savā projektā. Piemēram, tādas metodes kā Unity Ads, Analytics un pirkumus lietotnēs.

- Skaņas

  Ir pieejama skaņu failu bibliotēka, kuru varat izmantot, lai bagātinātu projekta lietotāja pieredzi.

- Veidnes

  Sadaļa Veidnes ļauj lejupielādēt dažādas apmācības un starta pakotnes, lieliska sadaļa iesācējiem.

- Rīki

  Šajā sadaļā ir pieejami noderīgi rīki, kas palīdz ar projekta izveidi. Ir plašs iespēju klāsts, sākot no AI līdz Visual Scripting.

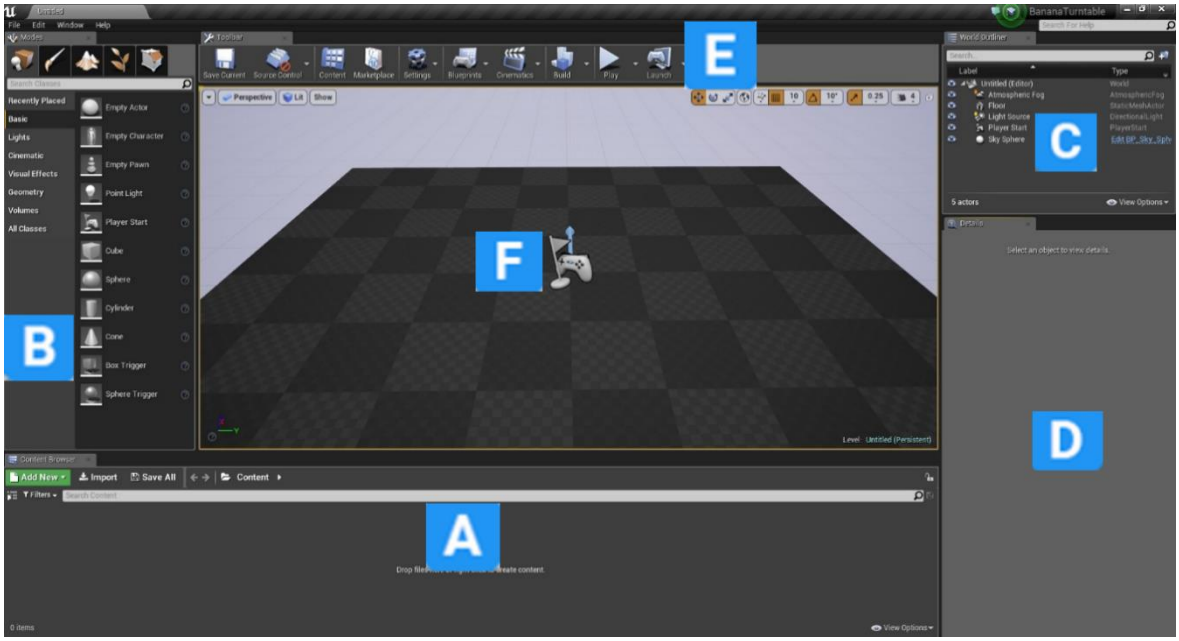
- VFX

Šajā sadaļā ir pieejami vizuālie efekti kā daļiņu efekti un ēnotāji. (Unity Technologies 2 2021)

1.2. UNREAL DZINIS

Unreal Engine ir spēļu dzinējs, kas palīdz veidot spēles. Unreal Engine sastāv no vairākiem komponentiem, kas darbojas kopā, lai vadītu spēli. Tā milzīgā rīku un redaktoru sistēma ļauj sakārtot savus aktīvus un ar tiem manipulēt, lai izveidotu spēli. Unreal Engine komponenti ietver skaņas dzinēju, fizikas dzinēju, grafikas dzinēju, ievadi un spēles ietvaru, kā arī tiešsaistes moduli. (Joanna Lee 2015)

Programmēšanas valodas: Unreal dzinis atbalsta C++ programmēšanas valodu, kā arī ir iespēja izmantot vizuālo skriptu izstrādi Blueprint.



Attēls 2 Unreal dziņa lietotāja UI (Tommy Tran 2017)

A – Satura pārlūks. Šajā panelī tiek parādīti visi jūsu projekta faili. To var izmantos, lai izveidotu mapes un kārtotu failus. Failus varat meklēt, izmantojot meklēšanas joslu vai filtrus.

B – Režīmi. Šis panelis ļauj izvēlēties starp rīkiem, piemēram, Ainavas rīks un Lapotnes rīks. Vietas rīks ir noklusējuma rīks. Tas ļauj jums ievietot daudz dažādu veidu priekšmetus savā līmenī, piemēram, gaismas un kameras.

C – Pasaules objektu logs. Parāda visus objektus pašreizējā līmenī. Sarakstu var sakārtot, mapēs ievietojot saistītos vienumus. Ir arī iespēja meklēt un filtrēt pēc veida.

D – Detaļu logs. Atlasītā objekta īpašības tiks parādītas šeit. Šo paneli var izmantot, lai rediģētu objekta iestatījumus. Veiktās izmaiņas ietekmēs tikai šo objekta gadījumu. Piemēram, ja jums ir divas sfēras, un mainât vienas lielumu, tas ietekmēs tikai atlasīto objektu.

E – Rīkjosla. Satur dažādas metodes. Visvairāk tiks izmantota Play poga.

F – Skats. Tas ir līmeņa skats. Ir iespējams paskatīties apkārt, turot ar peles labo pogu un pārvietojot peli. Lai pārvietotos, jātur labā peles poga jānoklikšķina un jāizmanto W,A,S,D taustiņus. (Tommy Tran 2017)

Unreal dziņa izmaksa: ir pieejami divas licences.

1. Publishing licence
2. Creators licence

Abas licences ir bezmaksas, bet Publishing licence atļauj veidot projektus kā spēles, kuras var publicēt un ar kurām ir iespējams pelnīt naudu, ja ienākumi pārsniedz $1000000 Unreal dzinis iekasē 5% autoratlīdzības. Creators licence nodrošina iekšējo projektu veidošanu bez maksas, bet ar šo licenci nav atļauts Veidot spēles un citus interaktīvus produktus, kurus iespējams publicēt.

Unreal Engine Marketplace ir e-komercijas platforma, caur kuru satura veidotāji, kas izmanto UE4, sazinās ar izstrādātājiem, nodrošinot spēli ar gatavu saturu, kodu.

Ir iespējam kļūt par izdevēju un iesniegtu savus produktus, izmantojot izdevēju portālu, un, ja tas tiks apstiprināts, tiks saņemti 88% no bāzes cenas par katru publicēto produktu pārdošanu. Saturs ir sadalīts vairākās sadaļās:

- 2D saturs

  Tēli, ikonas lietotāja interfeiss, vide, materiāli.
- Skaņa

  Fona mūzika skaņas efekti.
- VFX

  Vizuālie efekti
- 3D saturs

  Tēli, tekstūras, vide, animācijas, ieroči.

1.3. DARBĀ IZMANTOTĀ DZIŅA IZVĒLE

Abi spēļu dziņi gan Unity, gan Unreal ir labas izvēles spēles veidošanai, abiem dziņiem ir lielas kopienas un ar abu dziņu palīdzību ir veidotas labas un veiksmīgas spēles. Unity Asset store ir ievērojami lielāks nekā Unreal, kas dod priekšrocību mazām komandām, vai šajā gadījumā autoram. Abi dziņi ir ļoti līdzīgi, un izvēle beigās ir subjektīva, nevis pamatota ar kādu lielu plusu vai mīnusu. Autoram ir lielāka saskarsmes ar Unity, Unity ir viens no iemesliem, kāpēc autors interesējas par spēles izveidi, un arī viens no pirmajiem dziņiem par ko autors uzzinājis, tāpēc autors ir izvēlējies izveidot spēli, izmantojot Unity dzini. Autors noteikti nākotnē kādam projektam izmantos Unreal dzini, lai saprastu dziņu atšķirības.

2. SPĒLES IZVEIDES PROCESS

Spēles izstrādes process un spēles noformēšanas process ietver trīs galvenās fāzes.

1. **Pirms-ražošanas posms.**

Plānošana. Šajā posmā videospēles idejai ir jāprecizē, jāizvēlas spēles galvenā doma un virziens. Tajā ir izklāstītas tādas būtiskas lietas kā budžets, mērķauditorija, vai spēle būs 2D vai 3D, kādi būs varoņi un kurā platformā būs spēle. Šī ir plānošanas posma pirmā daļa un saknes, no kurām augs katra videospēle.

Visa pirms ražošanas posmā izveidotā pamatinformācija ir iekļauta Spēles noformēšanas dokumentā, kas vadīs visu komandu visā spēles izstrādes procesā un uzturēs sākotnēji izveidotā projekta redzējumu.

Prototipu veidošana. Spēļu prototipu izstrāde spēļu izstrādē ir būtiska, jo tā var ietaupīt daudz izšķērdēta laika un naudas. Šajā posmā jūs varat pārbaudīt spēles funkcionalitāti, lietotāja pieredzi, spēles gaitu, mehāniku un mākslas virzienu. Dažreiz spēle neiztur šo pārbaudi - tāpēc ir svarīgi iziet šo posmu. Viens padoms ir lūgt kādu citu pārbaudīt jūsu prototipu, jo jūs varētu palaist garām noteiktas lietas. (Starloop Studios 2020)

2. **Ražošanas posms**

Šis spēles izstrādes posms ir vissarežģītākais un izaicinošākais, taču šeit notiek burvība, un spēles ideja tiek iedzīvināta. Šajā posmā katram komandas loceklim ir labi izveidota loma. Projekta vadītājs vai spēļu producents ir atbildīgs par labu koordināciju starp komandas locekļiem. Viņam ir jānodrošina projekta vienmērīga norise, jāparedz un jāatrisina riska situācijas. Spēļu izstrādātāji raksta tūkstošiem rindiņu koda, lai katrs spēles saturs tiktu atdzīvināts. (Starloop Studios 2020)

3. **Pēc-ražošanas posms**

Apkope Pat ja ražošanas posms ir pabeigts, process tiek turpināts ar spēles uzturēšanu gadījumā, ja parādās kļūdas (parasti tas notiek diezgan bieži, bet tie tiek atrisināti apkopes posmā). Mārketings Arī videospēļu mārketings notiek visā tās ražošanas laikā un turpinās vēl kādu laiku pēc tās efektīvas izlaišanas. (Starloop Studios 2020)

3. LĪDZĪGAS VIDEOSPĒLES

Pirmās personas šāvējs (FPS) ir darbības videospēļu žanrs, kas tiek spēlēts no galvenā varoņa skatupunkta. FPS spēles parasti dod iespēju vadīt spēlētāju, kas sniedz skatu uz to, ko faktiskais cilvēks redzētu un darītu pašā spēlē.

FPS parasti parāda varoņa rokas ekrāna apakšdaļā, nēsājot līdzi jebkādu ieroci. Paredzams, ka spēlētājs virzīs sevi caur spēli, pārvietojoties uz priekšu, atpakaļ, uz sāniem utt.. Vadītāja kustības uz priekšu noved pie tā, ka spēlētājs virzās uz priekšu pa ainavu, parasti ar nelielu kreiso-labo šūpošanas kustību, lai pareizi simulētu cilvēka gaitu. Lai palielinātu reālisma līmeni, daudzās spēlēs papildus parastajiem skaņas efektiem ir iekļautas elpošanas skaņas un soļi. (Techopedia 2021)

3.1. SUPERHOT

Attēls 3 Superhot (Superhot Team 2021)

Superhot ir dzimis no game jam- 7 dienu FPS izaicinājuma, kurā izstrādātāji nedēļas laikā izveidoja pirmās personas šāvēju. Īss projekts izrādījās tik populārs, ka komanda to

aizveda uz Kickstarter, cerot to paplašināt pilnā spēlē. (Philip Kollar 2016)

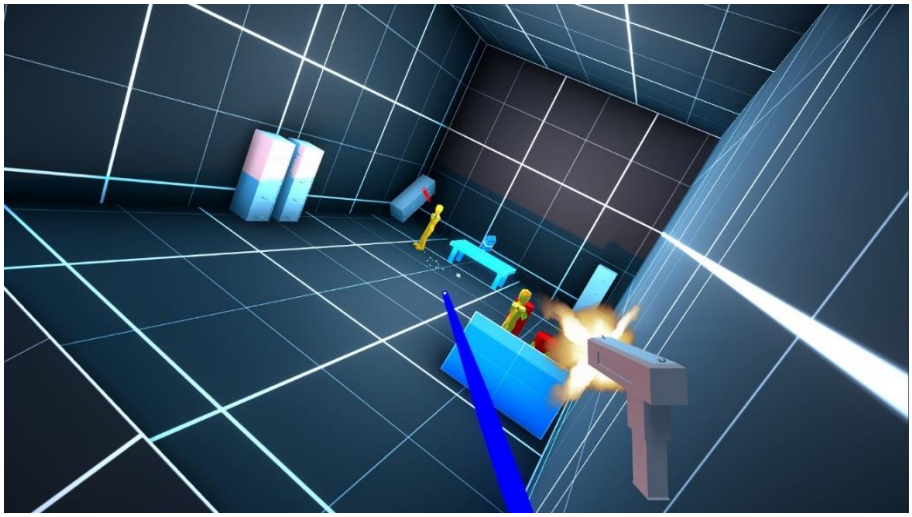
Attēls 4 Superhot spēle darbībā (Léon Othenin-Girard 2017)

Superhot spēle ir minimālisma šāvēja, kas piedāvā vairākas īsas un elegantas darbības sērijas, kuru laikā AI uzbrūk spēlētājam ar šaujamieročiem. Kā parasti pirmās personas spēlētājs ir bruņots ar ieroci. Superhot atšķirība no citām spēlēm ir tāda, ka spēlētājs spēj pārtraukt laiku, pārstājot kustēties. Šajā nekustīgajā laikā ir iespējams mērķēt ieročus. Kā arī spēlētājs spēj veikt fiziskus uzbrukumus ienaidniekiem un satvert ieročus gaisā, kombinējot šīs, kustības veidojas spēles ritms. (Colin Campbell 2015)

3.2. KARLSON

Attēls 5 Karlson (Dani 2021)

KARLSON piedāvā unikālu un uz prasmēm balstītu kustību, kas ir iedvesmota no klasiskajiem FPS. Spēles mērķis ir savākt piena paku līmeņa baigās spēlei ir vairākas mehānikas, kā skriešana pa sienai, šaušana sevis palaišana ar sprādzienu palīdzību, kā arī NPC pretinieki, kuru aizsargā piena paku. Šī spēle ir viena cilvēka projekts, kurš veidots ar Unity. (Dani 2021)



Attēls 6 Karlson spēle darbībā (Dani 2021)

4. SPĒĻU IETEKME UZ CILVĒKU

Ļoti bieži dzirdam pretrunīgus ziņojumus par to, kā videospēles ietekmē mūsu smadzenes. Dažādi pētījumi noved pie dažādiem secinājumiem - viena pētījuma rezultātā konstatē, ka videospēles palīdz mums mācīties, bet cita - videospēles padara jauniešus agresīvākus. Spēles bez šaubām ietekmē mūsu smadzenes, nav ne priekšlikums, ne noraidījums tam, ka spēlēm var būt gan pozitīvas, gan negatīvas sekas. Ir pierādīts, ka gan vardarbīgas, gan nevardarbīgas videospēles ietekmē spēlētāju radošumu. Eiropas Parlaments ir apspriedis, vai ierobežot bērnu piekļuvi videospēlēm.(Nicholas D.Bowman 2015)

Videospēles ir sava veida dabiski skolotāji. Tās sniedz tūlītēju atgriezenisko saiti par spēlētāja panākumiem, izdalot pastiprinājumus un sodus, palīdz mācīties dažādos tempos un piedāvā iespējas praktizēties līdz meistarībai un pēc tam līdz automātiskumam. Vairāki pētījumu virzieni liecina, ka videospēļu spēlēšana var dot dažāda veida priekšrocības, piemēram, Nature Neuroscience parādīja, ka darbības spēles var uzlabot pieaugušo spējas veikt smalkas atšķirības starp dažādiem pelēkajiem toņiem (to sauc par kontrasta jutību), kas ir svarīgi tādām aktivitātēm kā braukšana naktī, kā arī pētījumi pierāda to, ka cilvēkiem attīstās savstarpējās sadarbības prasmes.(Douglas A. Gentile, Ph.D 2009)

Literatūras apskats, ko veica Kvavaiders et al., radīja visaptverošu tabulu, kas savienoja dažādus videospēļu veidus ar to (negatīvo un pozitīvo) efektu. Protams, gan pozitīvā, gan negatīvā ietekme būs atkarīga no mainīgajiem lielumiem, piemēram, indivīdu personības, cik ilgi viņi spēlē vai cik bieži, vai no viņu emocionālā līdzsvara un miega laika. (Youmatter 2020)

Game Type	Benefit	Impact on the players
Shooter/Action	Cognitive skills	Faster and more accurate attention allocation
		Higher spatial resolution in visual processing
		Enhanced mental rotation abilities
Strategic		Allocate resources more efficiently
		Filter out irrelevant information more effectively
Role-playing		Enhance memorization, analytical skills, past experience, and intuitions
		Improve problem-solving skills, collaboratively
Strategic	Motivational Benefits	Increase the intelligence and abilities,
		Resilience in the face of failure
		Develop an incremental theory of intelligence,
		Use failure as motivational tools to engaged and bolster one's efforts
		Effective for "training" new behaviours.
		Lead to lasting educational success.
Puzzle	Emotional Benefits	Mood management, enhance the positive feelings.
		Sources of inspiration and connectivity.
		Build social relationships.
playing		Increase commitment and achievement, and higher self-esteem.
		Promote relaxation, and ward off anxiety
		Dealing with frustration and anxiety in adaptive ways
Cooperative-based	Social Benefits	Immediate, short-term cooperative play
		Short and long term effects on "helping" behaviours
		Reduces feelings of hostility
Prosocial		Decrease players' access to aggressive cognitions
		Increases subsequent prosocial, cooperative behaviour
Role-playing		Increase group organization and leadership skills
		Rapidly learning social skills and prosocial behaviours

Tabula 1 Spēles iespaids uz spēlētāju

(Quwaider et al.)

Tādējādi kooperatīvajām spēlēm, kā arī sociālajām un lomu spēlēm ir cieša saikne ar sociālajiem ieguvumiem. Tie mazina naidīguma sajūtu, mazina kognitīvo agresiju, palielina sadarbības uzvedību un veicina ātru sociālo prasmju apguvi. (Youmatter 2020)

PRAKTISKĀ DAĻA

Projektā tika veidota pirmās personas spēle ar šaušanas elementiem, spēles izveidei tika izmantots Unity3D dzinis. Spēles mērķis ir iznīcināt mērķus pēc iespējas ātrāk, iegūstot labāku rezultātu, patērējot pēc iespējas mazāku laiku mērķu sašaušanai. Spēles personāžu ir iespējams vadīt – spēlētājam jāpārvietojas un jāiznīcina mērķi, tos sašaujot.

1. IZMANTOTĀS UNITY IEBŪVĒTĀS METODES

Unity iebūvētās metodes ir īpašas metodes, kas pieejamas, lietojot Unity dzini. Šīs metodes parasti veidotas ar mērķi atvieglot spēles izstrādātāja darbu, lai izstrādātājs varētu fokusēties uz spēles īpašām funkcijām un neuztraukties par pamatu.

Start() metode tiek izsaukta tieši vienu reizi pirms Update() metodes, tieši objekta izveides brīdī vai skripta pirmās izsaukšanas mirklī.

Update() metode tiek izsaukta vienreiz kadrā, šī metode izmantota gandrīz katrā spēles skriptā.

Invoke() metode izsauc norādīto metodi pēc izvēlētā laika.

Transform.position komponenti izmanto, lai iegūtu spēles objekta atrašanās vietu spēles vidē, un mainot transform.position vērtību spēles objektu var pārvietot.

Transform.forward komponente atgriež normalizētu vektoru, kas norāda uz objekta z asi (uz priekšu vai atpakaļ) mainot transform.forward vērtību objektu iespējas kustināt, ignorējot tā rotāciju.

Instantiate() metode izveido eksistējoša objekta kopiju.

Destroy() metode izdzēš noteikto spēles objektu.

OnCollisionEnter() metode tiek izsaukta, kad objekts saskaras ar citiem objektiem.

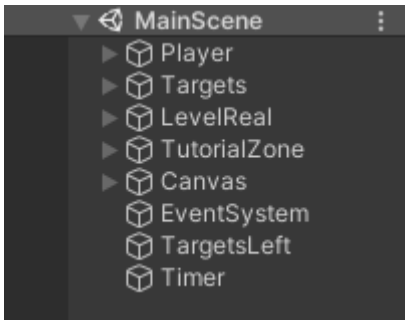
Move() metode pārvieto spēles objektu dotajā virzienā.

Physics.CheckSphere() metode atgriež “true”, ja kāds cits objekts pārklājas ar sfēru.

2. SPĒLES AINAS

Spēlei ir divas ainas “MainScene” aina un “MainMenu” aina. “MainMenu” aina ir tikai priekš galvenās izvēlnes un sastāv no kameras un divām UI pogām “Play” un “Quit”. Ar “Quit” pogas palīdzību, iespējams izslēgt spēli. Ar “Play” pogas palīdzību, iespējams, pāriet uz “MainScene” ainu, kurā ir visa spēle. “MainScene” ainā atrodas spēlētājs, visi

mērķi, viss līmenis un visi papildu skripti. “MainScene” aina sadalīta piecās galvenajās daļās.

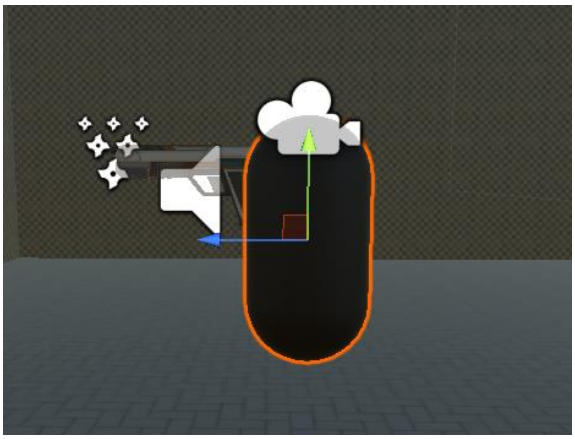


Attēls 7 MainScene aina

“Player” daļa satur sevī spēlētāju ar ieroci, “Targets” daļa satur sevī visus spēles mērķus, “LevelReal” daļa satur visu fizisko līmeņa izkārtojumu, “TutorialZone” daļa satur sākuma zonas fizisko izkārtojumu ar portālu un “Canvas”, kas satur UI elementus kā tēmekli ekrāna centrā, laika tekstu un mērķu tekstu, kā arī pauzes ekrānu. Sākoties “MainScene” ainai spēlētājs atrodas sākuma zonā un lai spēlētājs pārietu uz spēles līmeni, tam jāiet cauri portālam.

3. SPĒLĒTĀJA UN KAMERAS IZVEIDE

Spēles personāžs ir 3D kapsula, kam dots melns materiāls. Tā kā spēles darbība notiek no pirmās personas, spēlētājs savu ķermeni neredz. Tika izvēlēta tieši kapsula, jo tās forma ir apaļa un apaļš collider palīdz spēlētājam pārvietoties pa slīpām virsmām vai kāpnēm.



Attēls 8 Spēles personāžs

Personāžam tika pievienots Character Controller, kura galvenā priekšrocība ir kontroles daudzums, kas, dots par to, kā spēlētāja kontrolieris un personāžs mijiedarbojas ar spēles vidi, bet viens no trūkumiem ir tas, ka praktiski viss tas būs jāprogrammē. Kā arī

kustības skripts PlayerMovement.cs, kas ir atbildīgs par spēles personāža kustību. Kamera ar peles kustības skriptu MouseLook.cs, kas ir atbildīgs par kameras kustību un spēles personāža rotāciju atkarīgi no peles kustības

Spēlētāja kustība notiek ar metodi Move(), kurai tiek padots vektors, kurā atrodas informācija par piespiestajiem taustiņiem. Vektors tiek reizināts ar ātrumu, un "DeltaTime", lai kustības ātrums nebūtu atkarīgs no spēles kadru ātruma.

Lai spēlētājs spētu palēkties tam, tika pievienots "GroundCheck" objekts, kuru pārbaudot ar Physics.CheckSphere() metodi var noteikt, vai spēlētājs atrodas saskarē ar zemi.

Metode Physics.CheckSphere atgriež vērtību patiess, ja kāds objekts pēc definējuma saskaras ar sfēru. Sfēra šajā gadījumā ir tukšs GroundCheck objekts, kurš atrodas spēlētāja kapsulas lejā, vietā kur būtu pēdas. Visiem spēles vides objektiem ir piešķirts savs slānis - objektiem, kuri ir uzskatāmi, par zemi tiek piešķirts "Ground" slānis, ko arī padod Physics.CheckSphere metodei.

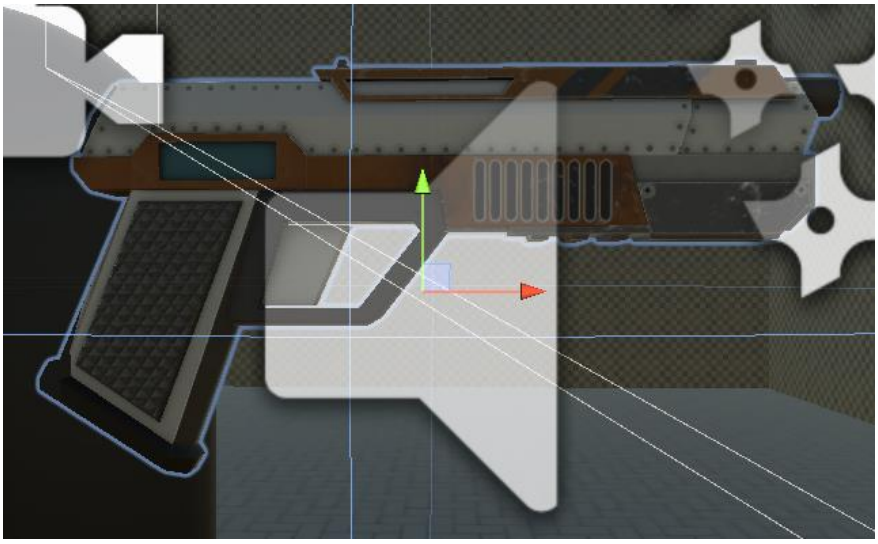
Lai pēc palēkšanās spēlētājs atgrieztos uz zemes, tika izveidota gravitācija, kura palielinās atkarīgi no tā, cik ilgi spēlētājs nepieskaras zemei un uzreiz pēc saskarsmes ar zemi tiek samazināta uz nulli.

Lai spēlētājs spētu skatīties apkārt kamerai, tika pievienots peles kustības skripts. Skripta sākumā kursors tiek fiksēts ekrāna centrā. Skatīšanās notiek, rotējot spēlētāju kopā ar kameru pa vertikālo asi, ja peli kustina horizontāli, un rotējot pašu kameru horizontāli, ja pele tiek kustināta vertikāli. Ja spēlētājs tiktu rotēts pa horizontālo asi kopā ar kameru, tad viss spēlētāja modelis griestos pa vidus asi un 'kūleņotu'.

Skriptā arī ir peles jūtības maiņas iespējas - spiežot '=' taustiņu peles jūtību iespējams palielināt un spiežot '-' taustiņu to pamazināt. Kā arī metode, kas maina peles jūtību un ir saistīta ar slīdni opciju logā.

4. SPĒLĒTĀJA IEROČA IZVEIDE

Spēles personāžam tika pievienots ierocis, kura modelis tika paņemta no Unity Asset Store. Ieroci veido četras daļas, kuras ir atdalītas, lai tām būtu iespējams izveidot animācijas.



Attēls 9 Ierocis

Ierocim tika pievienota kamera, kura redz tikai pašu ieroci, tādējādi risinot ieroča iegrimšanu sienās. Tā kā kamera redz tikai ieroci, tas nekad nepazūd no ekrāna.

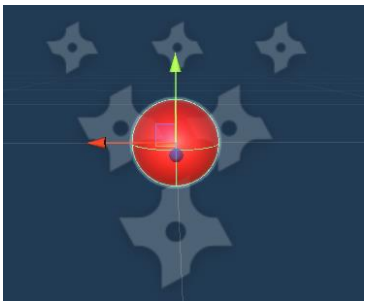
Šaušanas skripta GunScriptBalls.cs sākumā Start() metodē tiek izveidots skaņas komponents, tiek sākta Invoke() metode, kas pēc trim sekundēm izsauc metodi canShoot(), kas nomaina mainīgā "shooting" vērtību uz "true". Šīs darbības tiek veiktas, jo Update() metodē tiek veikta pārbaude, vai spēlētājs piespiedis kreisās peles taustiņu, vai ir pagājis pietiekams laiks pirms pēdējā šāviena, vai laiks nav apstādināts un vai mainīgā "shooting" vērtība ir "true". Gadījumā, ja tiek spiests kreisās peles taustiņš, ir pagājušas 3 sekundes pirms spēles sākuma, ir pagājušas 0.25 sekundes pirms pēlējā šāviena un spēle nav nopauzēta, tiek izveidots lodes objekts ar Instantiate() metodes palīdzību, tiek palaista šāviena skaņa un šāviena uzliesmojums. Instantiate() metode izveido lodes kopiju un izveidotā lode ar transform komponentes palīdzību tiek pārvietota priekšā kameras objektam, kas ir ekrāna centrs. Kā arī tiek uzstādīts laiks "nextFire", kas nosaka nākamā iespējamā šāviena laiku.

.

5. INTERAKTĪVIE SPĒLES ELEMENTI

Interaktīvi spēles elementi papildina spēli un dod iespēju spēlētājam veikt dažādas funkcijas. Spēlētājs spēj izveidot lodes un iznīcināt mērķus, kā arī izmantot portālu visas šīs darbības veic pats spēlētājs, mijiedarbojoties ar spēles elementiem.

5.1. LODES OBJEKTS

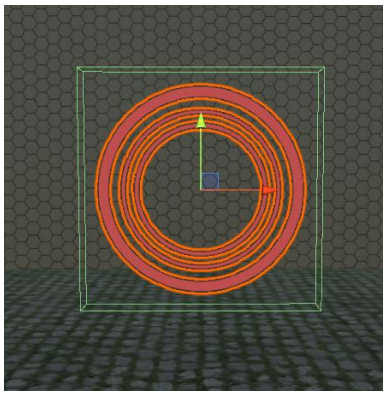


Attēls 10 Lode

Lode ir sfēra ar sarkanu materiālu un sarkanu spīdumu, materiāla un spīduma loma ir tikai izskats. Galvenais lodes komponents ir Sphere Collider, kas ļauj saprast, ar kādu objektu lode saskaras.

Lodes skripts BulletScript.cs ar transform komponentes palīdzību maina lodes pozīciju. Transform komponente ir iekš Upadet() metodes, jo lode maina pozīciju katru kadru tāpēc, spēlētājam izskatās, ka lode pārvietojas bez apstājas, bet tā katru kadru tiek mazliet pārbīdīta. Lodei saskaroties, ar kādu citu objektu tiek veikta pārbaude, izmantojot OnCollisionEnter() metodi. Gadījumā, ja lode saskaras ar citu lodi vai pašu spēlētāju, tā netiek dzēsta, bet jebkurā citā gadījumā lodes objekts tiek dzēsts. Īpašs gadījums ir lodes saskare ar mērķi, kura gadījumā tiek izsaukta noteiktā mērķa metode TakeDamage(). Lodei tās izveides mirklī iekš Start() metodes tiek piešķirts tās dzīves ilgums, kurš tiek pārbaudīts katru kadru un sasniedzot limitu lodes objekts tiek izdzēsts. Lodes tiek dzēstas, lai spēle pēc liela daudzuma ložu izveides nesāktu patērēt pārāk lielus resursus datoram.

5.2. MĒRĶIS



Attēls 11 Mērķis

Mērķim ir Box Collider, lai lode spētu saprast, kad tā saskaras ar mērķi un kad ir jāizsauc TakeDamage() metodi.

Skripts Target.cs piešķir mērķim dzīvības "health", kuras katram mērķim var būt atšķirīgas. Dzīvību skaits ļauj veidot mērķus, kuru iznīcināšanai būtu nepieciešamas vairākas lodes. Metode TakeDamege() mērķim atņem noteiktu dzīvību skaitu. Metode Die() tiek izsaukta TakeDamage() metodē, ja "health" vērtība ir vienāda vai mazāka par nulli. Die() metode izdzēš mērķi, un nomaina TargetsLeft.cs skripta mainīgā "targetCount" vērtība par mīnus viens.

5.3. PORTĀLS

Portāls ir taisnstūra paralēlskaldnis ar sarkanu materiālu.

Attēls 12 Portāls

Portāla funkcija ir pārvietot spēlētāju uz līmeņa starta pozīciju. Portāla galvenā komponente ir Box collider, kas ļauj uzzināt, kad objekts ar portālu saskaras.

Portālam ir skripts Teleport.cs, kam ir viena metode OnTriggerEnter(), kas pārvieto spēlētāju uz noteiktu pozīciju brīdī kad, kāds no spēlētāja objektiem pieskaras portālam. Tā kā lodes tiek veidotas ar ieroci un ierocis atrodas spēlētājam rokās, iešaujot lodi portālā, tas spēlētāju pārvieto. Spēlētājs tiek pārvietots uz iepriekš pievienotu spēles objektu, kurš atrodas līmeņa sākumā. Metode OnTriggerEnter() nomaina Timer.cs mainīgā "started" vērtību uz "true" un iestata Timer.cs "startTime" vērtību uz laiku, kurā spēlētājs izmantojis portālu.

6. CITI SKRIPTI

Šie skripti ir atbildīgi par spēles sākšanos un spēles beigām. Spēlei sākoties taimera skripts Timer.cs uzņem laiku un visas spēles garumā mērķu skaitītājs TargetsLeft.cs pārbauda atlikušo mērķu skaitu. Kad mērķi iznīcināti, tiek izsaukts spēles beigu skripts GameOver.cs un katras jaunas spēles sākumā šis cikls atkārtojas, bez šiem skriptiem spēlei nebūtu finiša.

6.1. TAIMERIS

Taimera skripts Timer.cs uzsāk laika atskaiti, kad spēlētājs ir sācis spēli, izmantojis portālu. Taimeris beidz darbību, kad visi mērķi iznīcināti un spēle uzveikta. Taimer.cs skripta ir viena metode Finish(), kas nomaina mainīgā "finished" vērtību uz "true" un izsaucot GameOver.cs metodi GOver() padodot esošo laiku. Skripts, katru kadru aprēķina patērēto laiku kopš spēles sākuma, noformatē šo laiku minūšu un sekunžu formātā un to iestata kā UI teksta objekta tekstu.

6.2. MĒRĶU SKAITĪTĀJS

Mērķu skaita skripts TargetsLeft.cs tā darbības sākumā iestata mērķu skaitu "targetCount", un spēles gaitā pārbauda mērķu skaitu. Gadījumā, ja mērķu skaits ir mazāks par viens, tiek izsaukta Timer.cs metode Finish(). Kā arī katru kadru tiek atjaunots UI mērķu skaits ar tekstu "Targets left:" un mērķu skaits, kas šobrīd nav iznīcināts.

6.3. SPĒLES BEIGU SKRIPTS

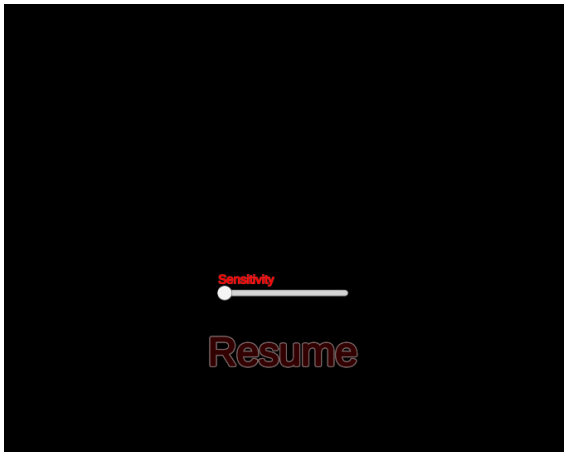
Spēles beigu skripta GameOver.cs skriptam tiek padots laiks no Timer.cs skripta. Skripta GOver() metodei tiek padots laiks un tā atbrīvo kursoru, ieslēdz spēles beigu ekrānu un parāda beigu laiku. Metode RestartButton() nomaina Timer.cs mainīgā "started" vērtību uz "false" un ielādē spēles sākumu. Metode MainMenu() nomaina kadru uz galveno izvēlni.

7. LIETOTĀJU SASKARNE

Spēlē ir šādi lietotāju saskarnes logi:

Attēls 13 Galvenā izvēlne

Galvenajā izvēlnē ir divas pogas "Play" un "Quit", un skripts MainMenu.cs, kas nodrošina pogu darbību. Skriptam ir divas metodes PlayGame() un QuitGame(). PlayGame() metode nomaina spēles ainu uz "MainScene", kur sākas spēle. QuitGame() metode izslēdz spēli.



Attēls 14 Pauzes logs

Pauzes logā iespējam mainīt peles jūtīgumu, ar slīdņa palīdzību. Pauzes loga skripts PauseScript.cs nodrošina spēles laika apstādināšanu nospiežot Escape taustiņu, skriptā ir divas metodes Resume() un Pause(), metodes ir pretējas. Pause() metode ieslēdz pauzes logu, atbrīvo kursoru, apstādina spēles laiku, izslēdz skaņu un nomaina mainīgā "GameIsPaused" vērtību uz "false". "GameIsPaused" mainīgais palīdz saprast, vai spēlētājs vēlas spēli nopauzēt vai atpauzēt. Metodē Resume() viss notiek otrādāk - tiek izslēgts pauzes logs,

kursors tiek fiksēts ekrāna centrā, laiks tiek atsākts, skaņa tiek ieslēgta, un mainīgā "GameIsPaused" vērtība nomainīta uz "false".

Attēls 15 Spēles beigu ekrāns

Izpildot spēles mērķi, tiek parādīts spēles beigu ekrāns, kura centrā ir uzrādīts iegūtais laiks un divas pogas.

Poga "Restart" izsauc spēles ainavu, un spēle sākas no jauna. Pogas "Main Menu" izsauc galvenās izvēlnes ainavu un spēlētāju aizved uz pirmo spēles logu.

8. SPĒLES IETEKME UZ CILVĒKU

Šaušanas spēles spēlētājam uzlabo uzmanības piešķiršanas prasmi, spēle liek izmantot savus resursus efektīvi tādējādi, uzlabojot resursu efektīvas izmantošanas prasmi, šaušanas spēles ar stratēģiskiem elementiem uzlabo atmiņu, atceroties iepriekšējo pieredzi, kā arī uzlabo problēmu risināšanas prasmes un attīsta analītiskās prasmes.

Darba autors uzskata, ka šī spēle var palīdzēt uzlabot atmiņu un stratēģisko domāšanu, atceroties, kur novietoti mērķi, un, izplānojot secību kādā, tos iznīcinot, ir iespējam iegūt labāko laiku. Spēlētājam jābūt ātram un precīzam, lai iegūtu ātrāko laiku. Katra kustība ir svarīga, testējot vairākus pieejas veidus un mērķu iznīcināšanas secības iespējam uzlabot laiku un rezultātu.

DARBA EKONOMISKAIS PAMATOJUMS

Pirmās personas šaušanas spēle, kas tika izstrādāta Unity3D vidē, un tās dokumentācija tika izstrādāta mācību nolūkos. Tas tika izstrādāts, lai autors un šī darba lasītājs varētu gūt ieskatu pirmās personas spēļu izstrādē Unity3D vidē. Projekts nenesīs peļņu, jo tas tika izstrādāts, lai autors iegūtu pieredzi un zināšanas par Unity3D dziņa iespējām un izmantošanu.

1. Tirgus konkurence

Pirmās personas šaušanas spēles liek spēlētājam piedzīvot visu no sava skatupunkta, kas ļauj labāk iejusties spēlē, tas ir viens no iemesliem, kāpēc FPS spēles ir viens no vispopulārākajiem spēļu žanriem. Izstrādātajai pirmās personas šaušanas spēlei ir milzīga tirgus konkurence. Lielākie FPS spēļu izstrādātāji ir - Electronic Arts, Ubisoft, CAPCOM, Valve corporation, Tencent, Epic Games. Protams, šī brīža stadijā autora spēle nav spējīga konkurēt, bet, ja spēle tiktu papildināta pievienojot jaunus līmeņus un mehānikas, konkurētspēja palielinātos.

2. Izstrādes izdevumi

Visi darba izstrādes izdevumi tiek iedalīti trīs daļās:

- Tehniskais nodrošinājums
- Programmatūra
- Cilvēkresursi

Kopējais darba izstrādes laiks, kas tika veltīts šim projektam, ir aptuveni 90 stundas. Šajā laikā tika veidota dokumentācija, izstrādāta programmatūra, un izpētīta Unity3D vide. Pieņemot, ka programmētāja alga ir aptuveni 20 eiro stundā, darba izmaksas sastāda 1600 eiro. Tomēr gada projekta izstrādes ietvaros par darbu samaksa netika maksāta, tādēļ reālo izdevumu nav.

Tabulā ir uzskaitīti izdevumi, kas radās, izstrādājot pirmās personas šaušanas spēli. Tā kā visa izmantotā programmatūra ir pieejama bezmaksas, un tehniskais nodrošinājums jau pirms projekta bija pieejams, tad reālu izstrādes izdevumu nav.

Resurss	Vienību skaits	Izmaksas	Skaidrojums
Tehniskais nodrošinājums:			
Portatīvais dators	1	600	Iegādāts pirms projekta veidošanas.
Interneta pieslēgums	1	0	Pieejamu nesaistīti ar projektu.
Programmatūra			
Unity3D	1	0	Bezmaksas spēļu izstrādes dzinis.
Microsoft Visual studio 2019	1	0	Bezmaksas licence.
Cilvēkresursi			
Darba stundas	80	1600	Aptuvenās izmaksas izmantojot programmēšanas pakalpojumus.

Tabula 2 Izstrādes izdevumi

3. Ieguvumi no projekta

Veidojot pirmās personas šaušanas spēli, autors ir ieguvis gan praktiskas, gan teorētiskas zināšanas par Unity3D spēļu dziņa izmantošanu. Autors uzzināja vairāk informācijas par spēļu izstrādi un iespējamā.

SECINĀJUMI

1. Darba sākumā tika izvirzīts mērķis izstrādāt pirmās personas spēli ar šaušanas elementiem un tas tika izpildīts. Darba mērķis tika sasniegts iepriekš, veicot detalizētu spēļu dziņu analīzi un izvēli.

2. Projektu iespējas uzlabot vai izmantot kā piemēru citu spēļu izveidei autors plāno papildināt spēli ar dažādiem līmeņiem un papildu funkcionalitāti.

3. Spēles izveides procesā tika secināts, ka spēles izveides process aizņem daudz laika.

4. Takā autoram tā bija pirmā spēle un projekts, izmantojot Unity3D dzini projekta izstrādes laikā, tika gūtas papildu zināšanas spēļu izstrādē un Unity3D spēļu dziņa vidē.

5. Projekta rezultātā tika izveidota pirmās personas šaušanas spēle ar Unity3D dziņa palīdzību. Spēlētājs spēj kustēties un šaut, spēles mērķis ir, pēc iespējas ātrāk iznīcināt mērķus iegūstot ātrāko laiku.

6. Veidojot projektu autoram, nācās izmantot GitHub, tāpēc autors ieguva zināšanas arī par GitHub lietošanu, kas ir ļoti pozitīvi ņemot vērā GitHub plašo pielietojumu IT jomā.

7. Izveidotā spēle var palīdzēt spēlētājam uzlabot domāšanu un atmiņu spēlētājs var izaicināt sevi iegūt jaunu laika rekordu izmēģināt vairākas stratēģijas un testējot vairākas pieejas un mērķu iznīcināšanas secības.

LITERATŪRA

1. Anurag, 30.03.2018, “13 Pros & Cons to Know Before Choosing Unity 3D “, https://www.newgenapps.com/blog/unity-3d-pros-cons-analysis-choose-unity/ Skatīts: 03.2021
2. Lisa Bowen, 01.02.2014, “Video game play may provide learning, health, social benefits, review finds”, https://www.apa.org/monitor/2014/02/video-game Skatīts: 03.2021
3. Nicholas D.Bowman, 2015, “Chapter 2 - The Impact of Video Game Play on Human (and Orc) Creativity “, https://www.sciencedirect.com/science/article/pii/B9780128014622000023 Skatīts: 03.2021
4. Colin Campbell, 16/06/2015, “Superhot is a whirling ballet of bullets“, https://www.polygon.com/2015/6/16/8788059/superhot-is-a-whirling-ballet-of-bullets Skatīts: 03/2021
5. Dani, “Karlson”, https://danidev.itch.io/karlson Skatīts: 03.2021
6. Douglas A. Gentile, Ph.D, 23.07.2009., ”Video Games Affect the Brain—for Better and Worse”, https://www.dana.org/article/video-games-affect-the-brain-for-better-and-worse/ Skatīts: 04.2021
7. Léon Othenin-Girard, 19.07.2017, “4 Superhot VR Tips I Wish I’d Known Before I Started Playing”, https://culturedvultures.com/superhot-vr-tips/ Skatīts: 04.2021
8. Philip Kollar, 25.02.2016, “SUPERHOT REVIEW”, https://www.polygon.com/2016/2/25/11094044/superhot-review-pc-windows-xbox-one Skatīts: 04.2021
9. Joanna Lee, 9.12.2015, ”An overview of Unreal Engine”, https://hub.packtpub.com/overview-unreal-engine/ Skatīts: 04.2021
10. Tony Ortega, 12/02/2019, “GAME DEVELOPMENT”, https://thegamingeek.com/game-development/ Skatīts: 04.2021
11. Starloop Studios, 23.09.2020, “Game Development Stages: How Video Games Are Created?”, https://starloopstudios.com/game-development-stages/ Skatīts: 04.2021
12. Superhot Team, “SUPERHOT”, https://www.kickstarter.com/projects/375798653/superhot Skatīts: 04.2021

13. Unity Technologies, 16.03.2021, “Unity’s interface”, https://docs.unity3d.com/Manual/UsingTheEditor.html Skatīts: 04.2021
14. Unity Technologies 2, “Quick guide to the Unity Asset Store”, https://unity3d.com/quick-guide-to-unity-asset-store Skatīts: 04.2021
15. Techopedia, “First Person Shooter (FPS)”, https://www.techopedia.com/definition/241/first-person-shooter-fps Skatīts: 04.2021
16. Tommy Tran, 17.02.2017, “Unreal Engine 4 Tutorial for Beginners: Getting Started”, https://www.raywenderlich.com/771-unreal-engine-4-tutorial-for-beginners-getting-started Skatīts: 04.2021
17. Alex Twofaced, 07.11.2019, “Chances of Your Game Becoming Successful”, https://www.gamedev.net/tutorials/business/production-and-management/chances-of-your-game-becoming-successful-r5246/ Skatīts: 04.2021
18. Jeff Ward, 29.04.2008, “What is a Game Engine?”, https://www.gamecareerguide.com/features/529/what_is_a_game_.php Skatīts: 04.2021
19. Youmatter, 04.02.2020, “Do Video Games Lead To Greater Violence? The Pros And Cons Of Video Games”, https://youmatter.world/en/violence-pros-cons-video-games/ Skatīts: 04.2021

PIELIKUMS I

PROGRAMMAS PIRMKODS

BulletScript.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class BulletScript : MonoBehaviour
{
   public float speed = 8f;
   public float damage = 10f;
   public float lifeDuration = 20f;
   private bool collided;
   private float lifeTimer;
   private void OnCollisionEnter(Collision collision)
   {
      Target target = collision.transform.GetComponent<Target>();
      if (target != null && !collided)
      {
         target.TakeDamage(damage);
         collided = true;
         Destroy(gameObject);
      }

      if (collision.gameObject.tag != "Bullet" && collision.gameObject.tag != "Player" &&
!collided)
      {
         collided = true;
         Destroy(gameObject);
      }
   }
   void Start()
   {
      lifeTimer = lifeDuration;
   }
   void Update()
   {
      transform.position += transform.forward * speed * Time.deltaTime;
      lifeTimer -= Time.deltaTime;
      if (lifeTimer <= 0f) {
         Destroy(gameObject);
      }

   }
}
```

GunScriptBalls.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class GunScriptBalls : MonoBehaviour
{
    public GameObject bulletPrefab;
    public Camera cam;
    public ParticleSystem muzzleFlash;
    public float fireRate = 0.25f;
    private float nextFire;
    private bool shooting = false;
    AudioSource shoot;

    void Start()
    {
        Invoke("canShoot", 3.0f);
        shoot = GetComponent<AudioSource>();
    }

        void Update()
    {
        if (Input.GetButtonDown("Fire1") && Time.time > nextFire && Time.deltaTime!=0
&& shooting) {
            nextFire = Time.time + fireRate;
            muzzleFlash.Play();
            GameObject bulletObject = Instantiate(bulletPrefab);
            bulletObject.transform.position = cam.transform.position + cam.transform.forward;
            bulletObject.transform.forward = cam.transform.forward;
            shoot.Play();
        }
    }

    public void canShoot() {

        shooting = true;
    }
            }
```

MouseLook.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class MouseLook : MonoBehaviour
{
   public float mouseSensitivity = 90f;

   public Transform player;

   float xRotation = 0f;
   void Start()
   {
      Cursor.lockState = CursorLockMode.Locked;
   }
   void Update()
   {
      if (Input.GetKeyDown("=")) {
         mouseSensitivity += 10;
         Debug.Log(mouseSensitivity);
      }
      if (Input.GetKeyDown("-") && mouseSensitivity>10)
      {
         mouseSensitivity -= 10;
         Debug.Log(mouseSensitivity);
      }
      float mouseX = Input.GetAxis("Mouse X") * mouseSensitivity * Time.deltaTime;
      float mouseY = Input.GetAxis("Mouse Y") * mouseSensitivity * Time.deltaTime;

      xRotation -= mouseY;
      xRotation = Mathf.Clamp(xRotation, -90f, 90f);

      transform.localRotation = Quaternion.Euler(xRotation, 0f, 0f);
      player.Rotate(Vector3.up * mouseX);
   }

   public void changeSensitivity(float _sensitivity)
   {
      mouseSensitivity = _sensitivity;
   }
         }
```

PlayerMovement.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class PlayerMovement : MonoBehaviour
{

    public CharacterController controller;
    public float speed = 12f;
    public float gravity = -9.81f;
    public float jumpHeight = 3f;

    public Transform groundCheck;
    public float groundDistance = 0.4f;
    public LayerMask groundMask;
    Vector3 velocity;
    bool isGrounded;
    void Start()
    {
    }
    void Update()
    {
        isGrounded = Physics.CheckSphere(groundCheck.position, groundDistance,
groundMask);
        if (isGrounded && velocity.y < 0) {
            velocity.y = -2f;
        }
        float x = Input.GetAxis("Horizontal");
        float z = Input.GetAxis("Vertical");

        Vector3 move = transform.right * x + transform.forward*z;

        controller.Move(move*speed*Time.deltaTime);

        if (Input.GetButtonDown("Jump") && isGrounded)
        {
            velocity.y = Mathf.Sqrt(jumpHeight * -2f * gravity);
        }
        velocity.y += gravity * Time.deltaTime;
        controller.Move(velocity * Time.deltaTime);
    }
            }
```

GameOver.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;
using UnityEngine.SceneManagement;

public class GameOver : MonoBehaviour
{
    public Timer Timer;
    public Text timeText;
    public void GOver(string time)
    {
        Cursor.lockState = CursorLockMode.None;
        gameObject.SetActive(true);
        timeText.text = "Your time: " + time;
    }

    public void RestartButton() {
        Timer.started = false;
        SceneManager.LoadScene("MainScene");

    }
    public void MainMenu() {
        SceneManager.LoadScene("MainMenu");
    }
        }
```

Teleport.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class Teleport : MonoBehaviour
{
    public Transform teleportTarget;
    public GameObject thePlayer;
    void OnTriggerEnter(Collider other)
    {
        thePlayer.transform.position = teleportTarget.transform.position;
        Timer.started = true;
        Timer.startTime = Time.time;
    } }
```

TargetsLeft.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;

public class TargetsLeft : MonoBehaviour
{
   public static int targetCount=5;
   public Text targetText;
   const bool end = false;
   public Timer Timer;
   private void Start()
   {
      targetCount = 5;
   }
   void Update()
   {
      targetText.text = "Targets left: " + targetCount;

      if (targetCount < 1)
      {
         Timer.Finish();
      }
   }
}
```

Target.cs

```
using UnityEngine;

public class Target : MonoBehaviour
{
   public float health = 10f;
   public void TakeDamage(float amount)
   {
      health -= amount;
      if (health <= 0f)
      {
         Die();
      }
      void Die()
      {
         TargetsLeft.targetCount -= 1;
         Destroy(gameObject);
      }
   }
}
```

Timer.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;
using UnityEngine.SceneManagement;

public class Timer : MonoBehaviour
{
    public Text counterText;
    public static float startTime;
    private static bool finished=false;
    public static bool started = false;
    public GameOver GameOver;
    private static string currTime;

    void Start()
    {
        startTime = Time.time;
        finished = false;
    }

    void Update()
    {
        if (finished) return;
        if (started) {
            float t = Time.time - startTime;
            string minutes = ((int)t / 60).ToString();
            string seconds = (t % 60f).ToString("f2");
            counterText.text = "Time: " + minutes + ":" + seconds;
            currTime = minutes + ":" + seconds;
        }
    }

    public void Finish()
    {
        finished = true;
        GameOver.GOver(currTime);
    }
 }
```

PauseScript.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class PauseScript : MonoBehaviour
{
    public static bool GameIsPaused = false;
    public GameObject pauseMenu;

    private void Update()
    {
        if (Input.GetKeyDown(KeyCode.Escape))
        {
            if (GameIsPaused)
            {
                Resume();
            }
            else {
                Pause();
            }
        }
    }
    public void Resume() {
        Cursor.lockState = CursorLockMode.Locked;
        pauseMenu.SetActive(false);
        Time.timeScale = 1f;
        GameIsPaused = false;
        AudioListener.volume = 1;
    }
    void Pause() {
        Cursor.lockState = CursorLockMode.None;
        pauseMenu.SetActive(true);
        Time.timeScale = 0f;
        GameIsPaused = true;
        AudioListener.volume = 0;
    }
}
```

MainMenu.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.SceneManagement;

public class MainMenu : MonoBehaviour
{
    public void PlayGame() {

        SceneManager.LoadScene("MainScene");
    }

    public void QuitGame() {

        Application.Quit();
    }
}
```

PIELIKUMS II

PROGRAMMAS PROJEKTĒJUMA APRAKSTS

2.1. Ievads

2.1.1. Dokumenta nolūks

Šis dokuments ir pirmās personas šaušanas spēles programmatūras projektējuma apraksts (turpmāk tekstā – PPA). Dokumentā ir definētas PPA prasības, kuras tiek izvirzītas pirmās personas šaušanas spēlei un paredzētas lietošanai sistēmas izstrādātājam, kā arī tās lietotājiem par pamatinformāciju sistēmas izstrādē.

2.1.2. Darbības sfēra

Programmatūras projekta darbības sfēra ir izveidot pirmās personas šaušanas spēli, kurā pēc iespējas ātrākā laika posmā ir jāiznīcina mērķi, kuri ir novietoti stratēģiskās pozīcijās. Sistēma paredzēta, lai tās lietotāji spētu attīstīt loģisko domāšanu un atmiņu.

2.1.3. Definīcijas, akronīmi un saīsinājumi

LVS	Latvijas valsts standarts
Modulis	Atsevišķa identificējama programmas daļa, kuru var autonomi izveidot un izmantot, lai atvieglotu programmu sastādīšanu.
Process	Sistemātiska operāciju izpilde kāda noteikta rezultāta iegūšanai.

2.1.4. Saistība ar citiem dokumentiem

Šis dokuments ir veidots pēc LVS 72:1996 “Ieteicamā prakse programmatūras projektējuma aprakstīšanai” standarta noteiktajām prasībām.

2.1.5. Programmatūras dzīves cikls

Programmatūras dzīves cikls sākās 2021. gada 9. februārī, kad pirmās personas šaušanas spēle tika sākta plānot. Šobrīd dzīves cikls vēl turpinās. Projekts tiek nodots 2021. gada 7. jūnijā, bet par pilnīgu programmatūras izstrādes noslēgumu spriest nav iespējams.

2.2. Projektējum dekompozīcijas apraksts

Sistēma ir sadalīta 4 galvenajos moduļos. Spēles varoņa modulī, Lodes un mērķa modulī, Spēles laika un beigu modulī un Galvenās izvēlnes modulī. Katram no šiem moduļiem ir vairāki procesi, kas tālāk tiek uzskaitīti.

2.2.1. Moduļu un procesu dekompozīcija

2.2.1.1 Spēles varoņa modulis

Tips: Modulis

Nolūks: Ļaut lietotājam vadīt spēles varoni, lai pārvietotos un iznīcinātu mērķus.

Funkcijas: Nodrošināt spēles varoņa pārvietošanos spēles pasaulē, iznīcinot spēles mērķus. Nodrošina spēles varonim iespēju šaut lodes, lai iznīcinātu mērķus.

Spēles varoņa modulis sastāv no šādiem procesiem:

Staigāšana

Tips: Process

Nolūks: Pārvietot spēles varoni spēles vidē uz priekšu vai sāniem, lai iznīcinātu mērķus.

Funkcijas: Nodrošināt spēles varoņa pārvietošanās virzienu. Pārvietošanās laikā tiek pārbaudīts vai nav saskarsme ar citiem spēles objektiem, kuriem nav iespējam iziet cauri.

Lēkšana

Tips: Process

Nolūks: Pārvietot spēles varoni spēles vidē uz augšu, lai piekļūtu mērķiem.

Funkcijas: Nodrošina spēles varoņa pārvietošanās virzienu. Pārvietošanās laikā tiek pārbaudīts, vai spēles varonis saskaras ar zemi un nav saskarsme ar citiem spēles objektiem, kuriem nav iespējams iziet cauri.

Šaušana

Tips: Process

Nolūks: Izveidot lodi, lai iznīcinātu spēles mērķus.

Funkcijas: Sniedz iespēju izveidot lodi, piespiežot kreiso peles taustiņu.

2.2.1.2. Lodes un mērķa modulis

Tips: Modulis

Nolūks: Nodrošināt objektu pārvietošanu un iznīcināšanu.

Funkcijas: Objektu pārvietošana un iznīcināšana.

Spēles objektu modulis satur šādus procesus:

Lodes pārvietošanās

Tips: Process

Nolūks: Palaist lodi taisnā līnijā uz priekšu uzreiz pēc tās izveidošanas.

Funkcija: Palaiž lodi taisnā līnija uz priekšu uzreiz pēc tās izveidošanas.

Lodes saskarsme

Tips: Process

Nolūks: Iegūt informāciju par objektu, ar kuru lode ir saskarsmē, iznīcināt lodi, ja tā ir saskarsmē ar objektu, un izsaukt funkciju mērķim, ja lode ir bijusi saskarsmē ar to.

Funkcija: Iznīcināt lodi saskarsmē ar citiem objektiem. Izsaukt funkciju mērķim, ja lode ir bijusi saskarsmē ar to.

Mērķa iznīcināšana

Tips: Process

Nolūks: Iznīcināt mērķi, ja ar to ir saskārusies lode.

Funkcija: Iznīcina mērķi, lodes un mērķa saskarsmes gadījumā.

2.2.1.3. Spēles laika un beigu modulis

Tips: Modulis

Nolūks: Uzņemt spēles laiku uzreiz pēc tās sākšanas un noteikt spēles beigu iestāšanos.

Funkcijas: Uzņemt spēles laiku. Noteikt spēles beigu iestāšanos.

Spēles laika un beigu modulis sastāv no šādiem procesiem:

Taimeris

Tips: Process

Nolūks: Uzsākt laika atskaiti uzreiz pēc tās sākuma un apstādināt, kad spēle beigusies.

Funkcija: Uzsāk laika atskaiti un pabeidz to spēles beigās.

Spēles beigas

Tips: Process

Nolūks: Veikt pārbaudi, vai visi spēles mērķi iznīcināti, ja spēles mērķi ir iznīcināti izsaukt spēles beigas.

Funkcija: Pārbaudīt atlikušo spēles mērķu skaitu un izsaukt spēles beigas, ja visi mērķi iznīcināti.

Pauze

Tips: Process

Nolūks: Apstādināt spēli pauzējot laiku un, dodot funkciju, spēlētājam mainīt peles jūtīgumu.

Funkcija: Apstādina spēles laiku, dod iespēju mainīt peles jūtīgumu.

2.2.1.4. Galvenās izvēlnes modulis

Tips: Modulis

Nolūks: Ļaut lietotājam izvēlēties starp dažādām iespējām, pirms uzsākt spēli.

Funkcija: parādā izvēlni, kas ļauj izvēlēties sākt spēli, pamainīt peles jūtīgumu un iziet no spēles.

Galvenās izvēlnes modulis sastāv no šādiem procesiem:

Sākt spēli

Tips: Process

Nolūks: Ļaut lietotājam uzsākt spēli.

Funkcija: Pāriet no galvenās izvēlnes loga uz spēles sākumu un uzsāk spēli.

2.2.2. Vienlaicīgo procesu dekompozīcija

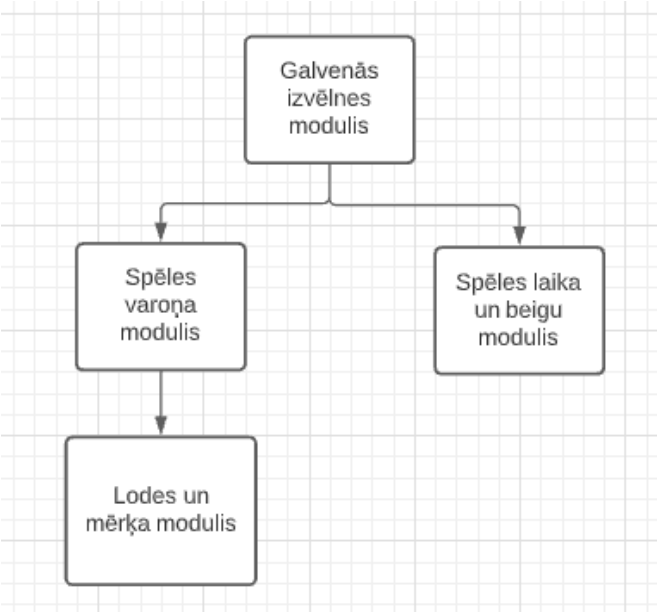
Vienlaicīgi vienmēr darbojas kāds no spēles varoņa moduļa procesiem un spēles laika un beigu moduļa procesiem, kā arī brīžos, kad spēlētājs šauj vienlaikus, darbojas arī lodes un mērķa modelis.

Kāds no spēles laika un beigu moduļa, kā arī kāds no spēles varoņa moduļa procesiem darbojas vienmēr, jo spēles varonim ir nepieciešam pārvietoties un laika atskaite neapstājas līdz spēles beigām. Brīžos, kas spēles varonis izveido lodi jeb šauj, tiek darbināti arī lodes un mērķa moduļa procesi, kas ļauj lodei pārvietoties un mērķim pazust.

2.3. Atkarību apraksts

2.3.1. Starp moduļu atkarības

Spēle ir sadalīta 4 moduļos. Bez spēles varoņa moduļa nevar darboties lodes un mērķa modulis, jo lodes tiek izveidotas ar varoņa palīdzību un bez lodēm mērķus nevar iznīcināt. Visi moduļi ir atkarīgi no galvenās izvēlnes moduļa tā kā, tajā ir jāveic izvēle spēlēt spēli, kas ļauj darboties pārējiem moduļiem. Detalizētāku moduļu atkarību skatīt 15. attēlā.



Attēls 16 Moduļu dekompozīcijas shēma

2.3.2. Starpprocesu atkarības

Galvenās izvēlnes process "Sākt spēli" ir tieši saistīta ar pārējiem procesiem, jo visi pārējie procesi var darboties tikai brīdī, kad spēle sākta. Lodes un mērķa procesi saistīti ar Spēles varoņa procesiem, jo tikai spēles varonis var izveidot lodes. Spēles laika un beigu procesi ir atkarīgi no Lodes un mērķa procesiem, jo tikai gadījumā, ja visi mērķi iznīcināti, spēle var beigties.

2.3.2.1 Galvenā izvēlne

Lai lietotājs spētu sākt spēles darbību, tam jāatrodas galvenajā izvēlnē un jāizvēlas sākt spēli. 19. attēlā ir redzama savstarpēji saistīto procesu aktivitāšu diagramma.

Attēls 17 Aktivitāšu diagramma galvenās izvēlnes procesiem

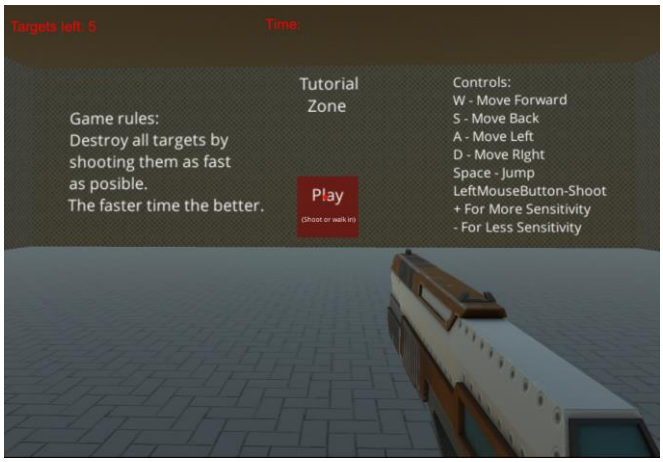
2.4. Saskarnes apraksts

2.4.1. Galvenās izvēlnes saskarne

Galvenās izvēlnes saskarne ir pieejama visiem spēlētājiem, kas atver spēli. Tās uzdevums ir ļaut lietotājam sākt spēli, mainīt iestatījumus, vai iziet no spēles. Skatīt 16. Attēlā.

2.4.2. Spēles vides saskarne

Spēles vides saskarne ir pieejama spēlētājiem, kuri galvenajā izvēlne ir, izvēlējušies sākt spēli. Tās uzdevums ir attēlot spēles logā informatīvu informāciju par spēles mērķi. Papildus sākuma zonā tiek attēlotas varoņa kontroles un spēles mērķis.



Attēls 18 Spēles vides saskarne

PIELIKUMS III

APLIECINĀJUMS par Autora mantisko tiesību nodošanu

APSTIPRINĀTS
ar Vidzemes Augstskolas rektora
2017.gada 17.maija rīkojumu Nr.7-r

APLIECINĀJUMS

par autora mantisko tiesību nodošanu

_____ (turpmāk – Darbs)

kvalifikācijas darbs/bakalaura darbs/maģistra darbs/gada projekts

*"*_____*",*

darba nosaukums

izstrādāts Vidzemes Augstskolas Inženierzinātņu fakultātē

Pamatojoties uz Autortiesību likuma 15.pantā noteiktajām mantiskajām tiesībām, kuras darba autors var nodot trešajām personām, **piekrītu**, ka mans Darbs tiek padarīts sabiedrībai pieejams bez maksas pilnā apjomā:

Nepiekrītu, ka manu Darbu padara sabiedrībai pieejamu ☐
Lūdzam norādīt pamatotu iemeslu:

_____

_____

_____

Darba autors: _____, _____, ___.___.201__.

parakts *vārds, uzvārds* *datums*

PIELIKUMS IV

APLIECINĀJUMS par darba atbilstību

APSTIPRINĀTS
ar Vidzemes Augstskolas rektora
2012.gada 11.aprīļa rīkojumu Nr.20-r

APLIECINĀJUMS
par darba atbilstību

……
atbilstošs ieraksts - kvalifikācijas darbs, bakalaura darbs, maģistra darbs, gada projekts

„……"
darba nosaukums

izstrādāts Vidzemes Augstskolas …….. fakultātē.
fakultāte

Ar savu parakstu apliecinu, ka darbs izstrādāts patstāvīgi un tajā ir atsauces uz visām izmantotajām citu autoru atziņām un datiem. Darbs izstrādāts saskaņā ar ViA ētikas pamatprincipiem, Studējošo akadēmiskās ētikas nolikumam un fakultātes metodiskajiem norādījumiem. Apzinos, ka plaģiāta konstatēšanas gadījumā darbs tiks noraidīts.

Iesniedzot darbu, uzņemos atbildību par jebkuras konfidenciālas informācijas, kas iegūta darba izstrādes gaitā, neizplatīšanu.

Darba autors: _____ / _____ / _____
autora vārds un uzvārds / *paraksts* / *datums*

Darbs iesniegts fakultātē _____ / _____ / _____
fakultātes vecākā speciālista vārds un uzvārds / *paraksts* / *datums*

Rekomendēju
darbu aizstāvēšanai _____ / _____ / _____
(aizpildīt, ja fakultātē noteikts) *darba vadītāja zinātniskais grāds, vārds un uzvārds* / *paraksts* / *datums*

Darbs aizstāvēts 201_.gada ___. _____ ar vērtējumu _____ (_____)
vērtējums cipariem *vērtējums vārdiem*

Valsts pārbaudījumu
komisijas priekšsēdētājs
(bakalaura un maģistra darbam)
vai studiju programmas
direktors *(gada projektam)* _____ / _____ / _____
valsts pārbaudījumu komisijas priekšsēdētāja vai studiju programmas direktora vārds, uzvārds / *paraksts* / *datums*