Proces izrade video igre koristeći Unreal Engine 4

Kuzminski, Bogdan

Sveučilište
Sjever

Završni rad br. 465/MM/2016

Proces izrade video igre koristeći Unreal Engine 4

Bogdan Kuzminski, 5413/601

Varaždin, listopad 2016. godine

Sveučilište
Sjever

Multimedija, oblikovanje i primjena

Završni rad br. 465/MM/2016

Proces izrade video igre koristeći Unreal Engine 4

Student

Bogdan Kuzminski, 5413/601

Mentor

Andrija Bernik, dipl. inf.

Varaždin, listopad 2016. godine

Predgovor

Industrija video igara danas je veća od filmske industrije sa gotovo 22 milijarde dolara godišnjeg prihoda. Upravo zbog toga za temu ovog rada je odabrana izrada video igre koristeći suvremeni softver i spajajući znanja iz raznih dijelova multimedije.

Sažetak

U ovom radu objašnjen je postupak kako napraviti jednostavnu video igre koristeći Unreal engine 4. Unreal engine 4 je softver koji se koristi za izradu video igra u 3D okruženju, a proizveden je od strane Epic Games-a. Unreal je besplatan i jedan od najljepših softvera za izradu video igra. Kod dizajniranja igre prvo izrađujemo scenu u koju zatim unosimo naše modele. Modeli mogu biti izrađeni u bilo kojem 3D softveru, nakon umetanja modela slijedi postavljanje istih u našu scenu. Scena predstavlja mapu ili svijet u kojemu će se igrač kretati. Nakon izrade svijeta slijedi izrada mehanike igre, Unreal daje dvije opcije programiranja, C++ ili Blueprints. Blueprints je poseban oblik vizualnog programiranja dizajniran posebno za Unreal Engine 4. On radi na principu spajanja blokova koji izvode određene radnje, upravo Blueprints je ono što odvaja Unreal Engine od konkurencije jer omogućava izradu video igra bez nekog znanja o programiranju. U ovom radu upravo radi toga je korišten Blueprints za programiranje mehanika igre. Na kraju rada objašnjen je postupak izrade grafičkog sučelja koristeći Unreal Motion Graphics (UMG).

Ključne riječi: Game engine, grafičko sučelje, blueprints, video igra, modeli

Sadržaj

1. Uvod 1
2. Unreal Engine 2
3. Zbrush i Maya 3
4. Izrada projekta 4
5. Pripremanje projekta 5
   5.1 Umetanje dodatnog sadržaja 6
6. Izrada mape 7
   6.1 Umetanje tla 8
   6.2 Skulpturiranje tla 9
   6.3 Stvaranje blend materijala 10
   6.4 Uređivanje novog materijala 11
   6.5 Dodavanje tekstura 12
   6.6 Bojanje mape novim materijalom 14
   6.7 Dodavanje drveća i ostalih modela mape 15
7. Dodavanje igrača i priprema za programiranje 17
   7.1 Dodavanje početne pozicije 18
   7.2 Izrada prvih klasa 19
8. Postavljanje sustava kontrolnih točaka 20
9. Početak programiranja sustava krugova 27
   9.1 Pisanje skripti u funkcije 30
   9.2 Stvaranje varijabli u kontroleru 34
   9.3 Završavanje kontrolera 40
   9.4 Dodavanje skripti u blueprint vozila 43
10. Izrada grafičkog sučelja 45
   10.1 Animacija 50
11. Sustav čestica 55
12. Zaključak 64
13. Literatura 65

1. Uvod

Cilj ovog rada je napraviti jednostavnu reli (eng. Rally) utrku koristeći Unreal Engine 4 te pripadajuću biblioteku dodataka (eng. marketplace).[21] Konačna verzija igre radit će na Windows sustavu no podržano je mnogo više sustava. Unreal Engine ima svoj sustav vizualnog programiranja nazvanog Blueprints, on je izrađen kako bi olakšao programiranje ljudima koji nemaju neko programersko znanje. U ovom radu također će se koristiti nekoliko integriranih modula za uređivanje poput Motion Graphics[16] koji služi za izradu korisničkih sučelja, Cascade[19] koji služi za uređivanje sustava čestica.

Verzija Unreala koja je korištena je 4.10.2. Sve verzije Unreala su besplatne te je u bilo kojem trenutno moguće skinuti neku stariju verziju ili noviju u slučaju da trenutno verziju želimo očuvati. Unrealova biblioteka se naziva Marketplace, ona sadrži sve modele, skripte, dodatke, zvukove, teksture i slično od kreatora Unreala, ali i od korisnika. Korisnici mogu koristiti biblioteku kao način da prodaju svoje radove, a moguće je pronaći i nekoliko besplatnih paketa koji su tu da olakšaju posao.

Unreal također osim biblioteke na kojoj dijele svoje službene radove i pakete nudi mogućnosti korištenja predložaka (eng. Template). Oni se učitavanju prilikom stvaranja novog projekta te dolaze sa početnim modelima i mehanikama igre koje zatim korisnik sam prilagođava svojem ukusu.

2. Unreal Engine

Unreal Engine je napravljen od strane Epic Games-a i prvi put je korišten u njihovoj igri Unreal 1998 godine. Prvotno je bio namijenjen za pucačine iz prvog lica no kasnije je počeo biti korišten u igrama raznih žanrova. Prva inačica je koristila svoj skriptni jezik UnrealScript koji je omogućio da svi koji žele stvaraju sadržaj za igru i mijenjaju je. [1]

Unreal Engine 4 je izašao 2012 godine te je donio ogromne promjene u samom softveru ali i u svijetu video igara zbog vrhunske vizualne kvalitete finalnih proizvoda i performansi. Unreal Engine 4 također je donio i mogućnost programiranja sa C++ ili Blueprints dok je igra pokrenuta. To omogućava programerima da vrlo brzo testiraju promjene nakon programiranja, u samoj igri bez dugih čekanja da se cijeli kod ponovno učita.

Unreal Engine 4 je od ožujka 2015. godine potpuno besplatan za sve kao i sve buduće verzije koje će dolaziti. Također i izvorni kod (eng. Source code) Unreal Engina također je u potpunosti dostupan javnosti .

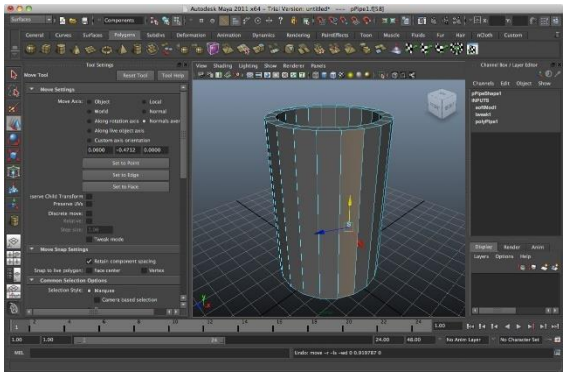
Unreal Engine 4 podržava sljedeće platforme:

- Microsoft Windows
- Linux OS X
- Xbox One
- PlayStation 4
- HTML5
- iOS
- Android
- VR (platforme virtualne stvarnosti)

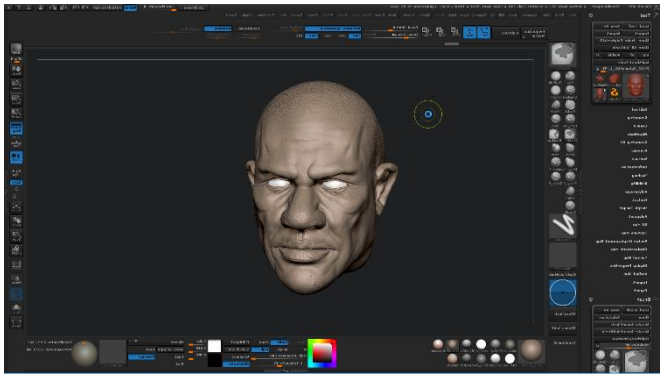
3. Zbrush i Maya

Za izradu 3D modela te animacija i njihovo testiranje prije nego se dodaju u samu igru danas su najpopularniji programi Zbrush te Maya. Maya je program za 3D modeliranje napravljen od strane kompanije Autodesk, za razliku od njegovog starijeg brata 3Ds Maxa, Maya je program u koji Autodesk želi ugurati što više mogućnosti i tako stvoriti program u kojem se može raditi sve što je vezano uz 3D modeliranje. U zadnjoj inačici tako su dodali mogućnosti osnovnog skulpturiranja. Najčešća uporaba Maye ipak je za 3D modeliranje i animaciju, ono što se prije moralo raditi u više različitih paketa softvera sada je moguće odraditi u jednome. Iako sama činjenica da Autodesk trpa sve mogućnosti u Mayu nije baš primamljiva nekome tko ne treba sve te mogućnosti, autorima koji rade samostalno jeftinije je kupiti Mayu i dobiti sve mogućnosti koje nudi umjesto kupovanja različitih softvera. [2]

Zbrush je program za 3D skulpturiranje napravljen od kompanije Pixologic. Zbrush danas predstavlja sam vrh 3D skulpturiranja i korišten je u gotovo svim kompanijama koje se bave bilo kakvim 3D projektima. Zbrush ima ogromnu bazu korisnika kao i profesionalnih kompanija koje koriste proizvod. Najviše je korišten za izradu vrlo detaljnih modela potrebnih u filmskoj i industriji video igara pošto obje industrije trebaju visokokvalitetne likove i modele. Zbrush omogućava umjetniku da u 3D prostoru koristeći kistove i sve njegove alate „nacrta“ model u vrlo visokoj rezoluciji sa fotorealističnih rezultatima, nešto što standardno modeliranje nemože. [3]



Slika 3.1 Autodesk maya



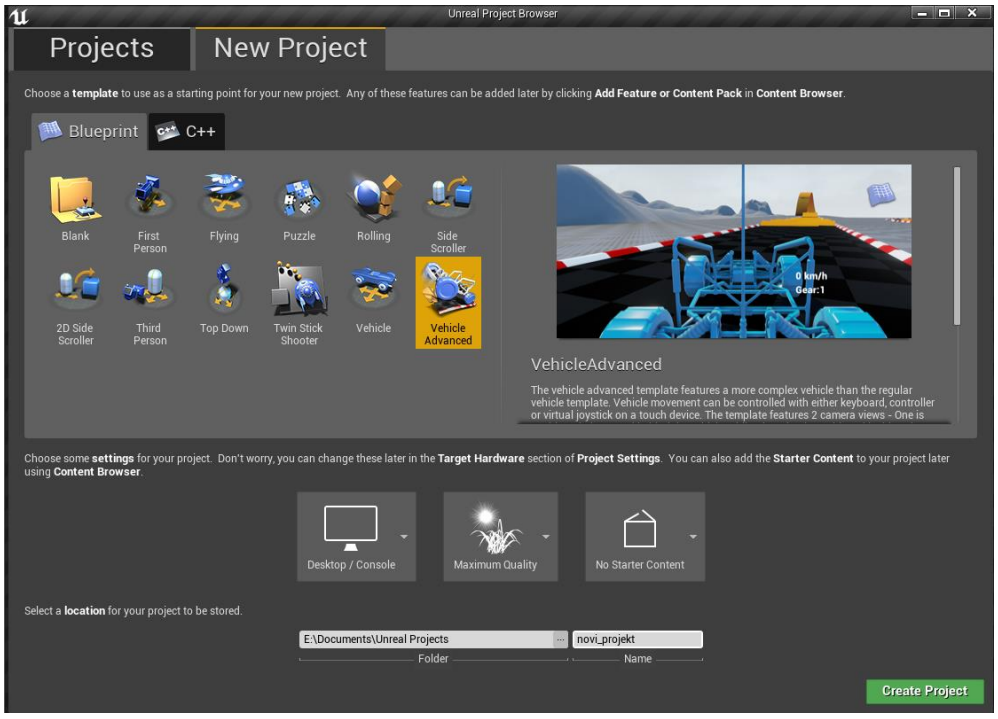
Slika 3.2 Zbrush

4. Izrada projekta

Kod prvog pokretanja programa pojavit će se prozor u kojemu se stvaraju novi projekti. Projekt može biti izrađen korištenjem Blueprints-a ili C++ programskog jezika. Osim izbora između načina programiranja Unreal nudi i opciju učitavanja predložaka u naš projekt [4]. Ponuđeni predlošci su:

- Pogled iz prvog lica
- Pogled iz trećeg lica
- Puzle
- Vozilo
- 2D
- Pogled odozgora

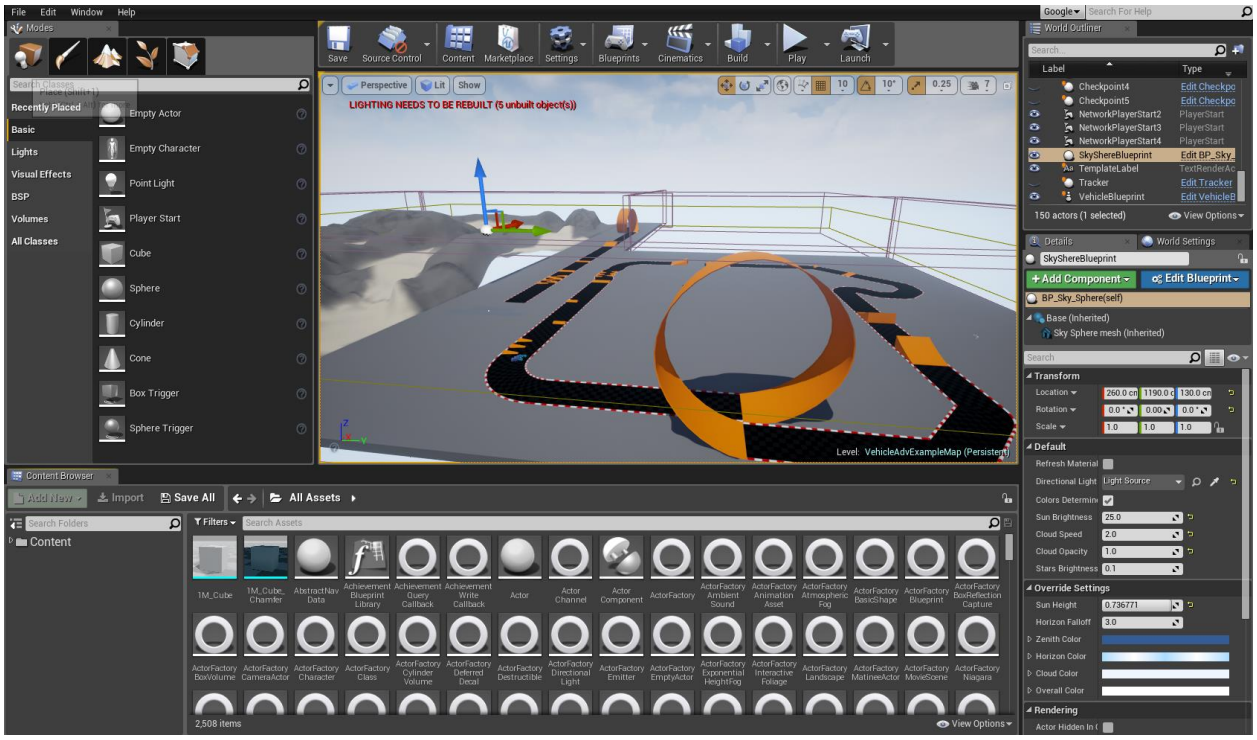
Neki od tih predložaka imaju i napredniju verziju istoga (eng. Advanced) koja nudi malo više opcija te im je početni model bolji i napredniji. U ovom radu izabran je predložak Vehicle Advanced. Također je odabrana i opcija „With starter content“[6] koja daje još nekoliko dodatnih modela i efekata čestica u projekt.



Slika 4.1 Prozor izrade novih projekata

5. Pripremanje projekta

Kada završi izrada projekta otvorit će se grafičko sučelje Unreal Engina i vidljivo je što se sve učitalo u projekt. U projektu se trenutno nalazi samo početni sadržaj koji je odabran kod kreiranja projekta.[4]

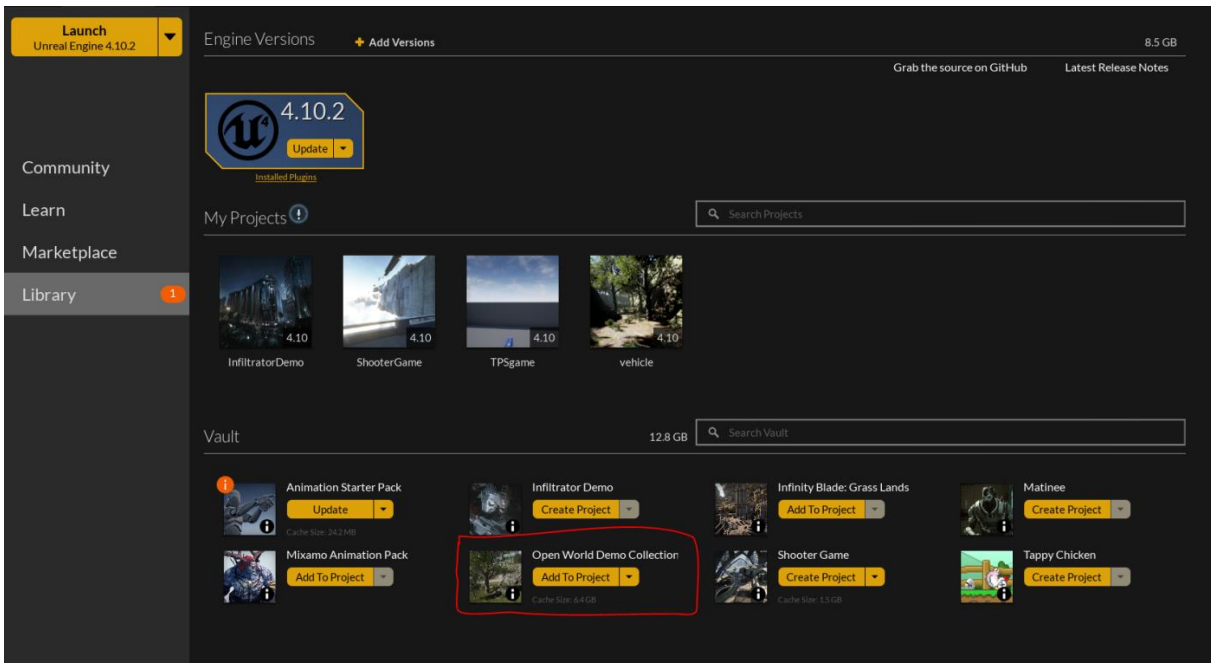


Slika 5.1 Izgled sučelja

Sa desne strane sučelja vide se osnovni modeli i opcije za geometriju, također i alate nužne za stvaranje mape i svijeta. Na dnu je popis svih modela i dodataka koji su trenutno u projektu, to uključuje 3D modele, njihove teksture, kosture za animacije, čestice, datoteke zvuka, video datoteke itd. Sa desne strane se nalazi popis svega što je trenutno u sceni kao i pripadajuće opcije za svaki selektirani objekt. Na samom vrhu se nalazi nekoliko opcija poput pokretanja igre i spremanja projekta.

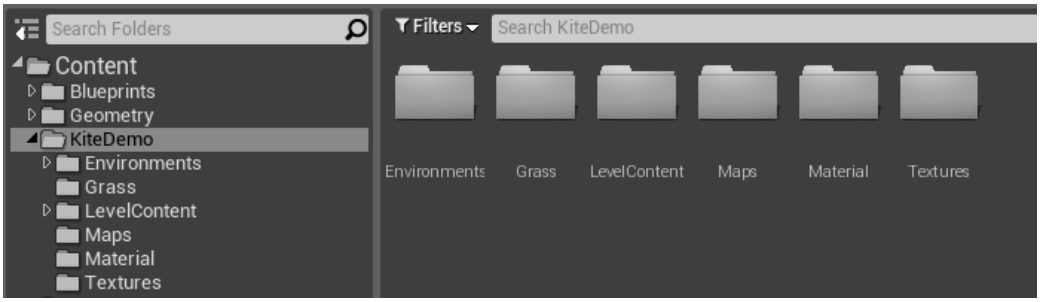
5.1 Umetanje dodatnog sadržaja

Kako bi se dodao sadržaj u projekt iz biblioteke potrebno se vratiti u Epic Launcher, to je program koji spaja sve proizvode kompanije Epic Games. Kod pokretanja Unreal Engina preko prečaca na radnoj površini prvo će se pokrenuti softver za lansiranje (eng. Launcher), a nakon što se on otvori otvorit će se i Unreal Engine. Nakon otvaranja biblioteke iz desnog izbornika odabire se paket koji treba dodati u projekt. U slučaju ovog rada to je paket Open world demo collection, te samo klikom na Add to project započeti će skidanje paketa na računalo. [5]



Slika 5.2 Dodavanje paketa u projekt

Nakon dodavanja paketa u projekt on će se nalaziti u donjem lijevom dijelu sučelja pod imenom Kite demo.

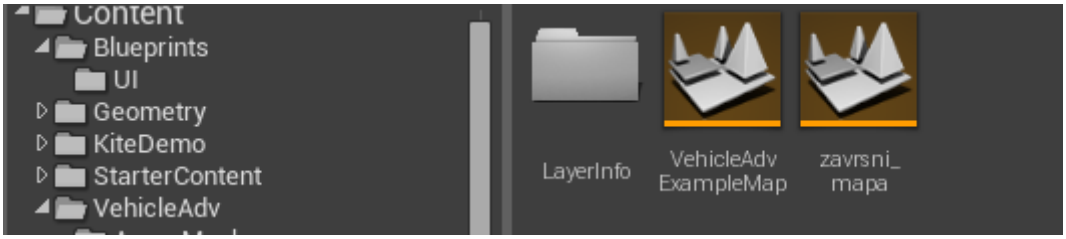


Slika 5.3 Izgled palete datoteka

6. Izrada mape

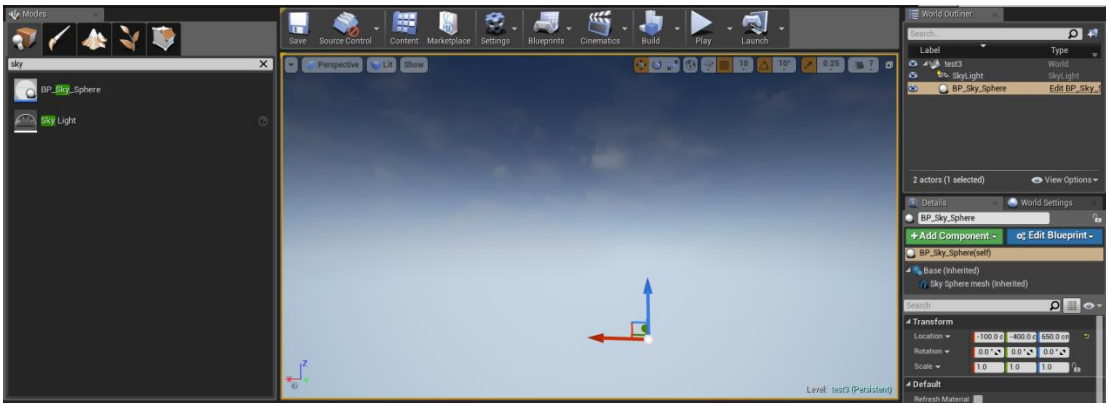
Prvi korak kod stvaranja igre bio bi izrada mape, može se koristiti i početna mapa za testiranje mehanika igre ali prije ili kasnije igra će se seliti u pravu mapu. Za izradu mape koristi se alat world builder, taj alat radi ogromnu površinu na kojoj se zatim može graditi mapa koristeći tehnike skulpturiranja i umetanja modela.

Za stvaranje nove mape u izborniku prvo u tražilicu se upiše maps, to će odvesti do mape u kojoj je spremljena početna mapa, nakon toga desnim tipkom miša klikne se u prazni prostor te odabere Level. To će stvoriti novu mapu koju je moguće nazvati po želji. Također istim postupkom napravljen je i folder nazvan LayerInfo, on će služiti za spremanje materijala koji će biti u mapi.



Slika 6.1 Izrada mape

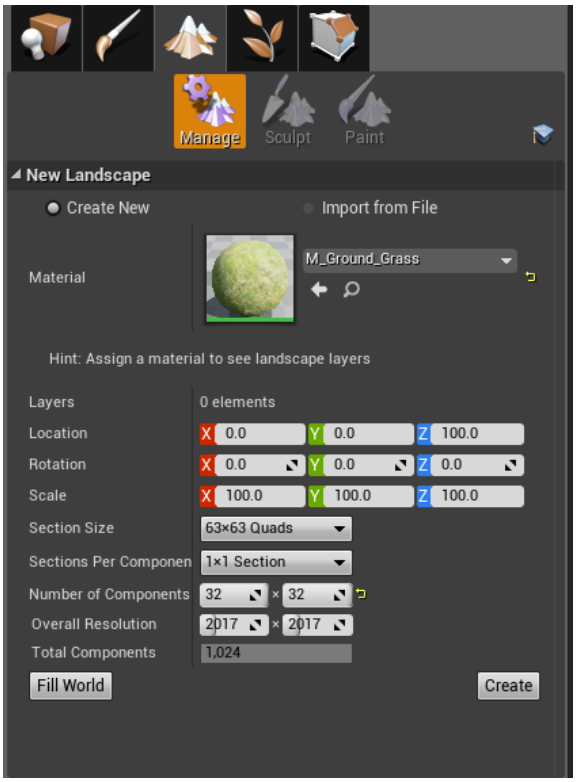
Nakon izrade mape potrebno je na nju dva puta kliknuti da se otvori. Otvorit će se prazna mapa, za lakše snalaženje dobro je prvo dodati neboiz lijevog izbornika te ga samo povući u centralni prozor.



Slika 6.2 Dodavanje neba

6.1 Umetanje tla

Sljedeći korak je dodati samu mapu tj. tlo, za to je potrebno kliknuti na ikonu planine u gornjem lijevom izborniku, otvorit će se graditelj svijeta koji ima nekoliko opcija koje se trebaju promijeniti.



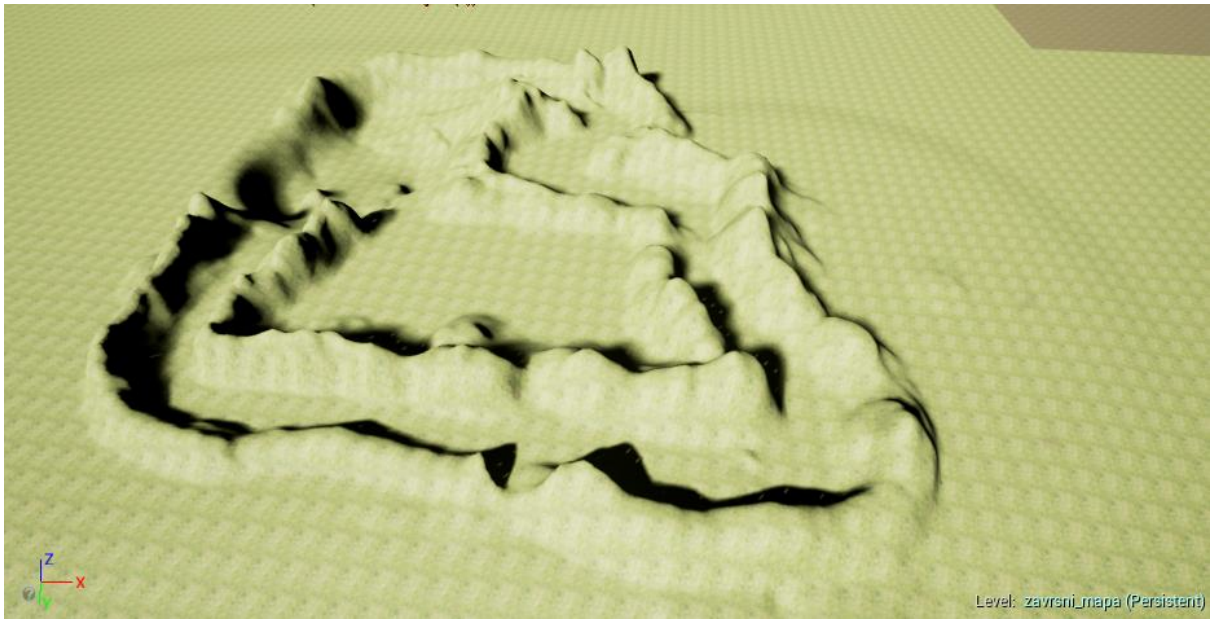
Slika 6.3 Prozor postavki tla

Prva i najbitnija promjena koju je potrebno napraviti je odrediti materijal, materijal je tekstura koja će se pojaviti po cijelom tlu. U ovom slučaju korištena je tekstura M_Ground_Grass koja dolazi besplatno uz Open world paket. Sljedeće opcije su veličina svijeta, koja može biti neke određene veličine ili jednostavno koristeći tipku Fill World popuniti cijeli svijet tom teksturom. Nakon pritiska tipke Create mapa će se popuniti odabranom teksturom. Nakon što se tlo učita dobro je dodati još jedno svijetlo u scenu, ovoga puta Directional light, koje također osvjetljava cijelu mapu te nudi opcije mijenjanja boje svijetla i intenzitet za postizanje ljepših efekata.

6.2 Skulpturiranje tla

Sljedeći korak je skulpturiranje tla koristeći dostupne kistove. Za pristup kistovima potrebno je kliknuti na Sculpt ikonu u gornjem lijevom izborniku koja se pojavi nakon pritiska na ikonu planine. U postavkama se odredi veličina i jačina kista te njegov tip pritiskom na tipku Sculpt tool. Držanje lijeve tipke miša tlo će uzdizati dok držanje tipke SHIFT i lijevog miša tlo će spuštati. Korištenjem te 2 metode izrađuje se mapa koja u ovom slučaju je zatvoreni krug gdje će se odvijati utrka.[8]

Slika 6.4 Izgled tla nakon kreiranja



Slika 6.5 Izgled završene mape nakon skulpturiranja

6.3 Stvaranje blend materijala

Unreal daje mogućnost izrade mješavine materijala (eng. Blend material) koji za razliku od običnih materijala ima nekoliko tekstura koje se preklapaju jedna s drugom. Ono što takav materijal omogućava je bojanje teksture na početni sloj bez grubih rubova na prijelazima između tekstura. Ta tehnika omogućava vrlo brzo teksturiranje velikih površina i uvijek se je moguće vratiti i nešto promijeniti. [9]

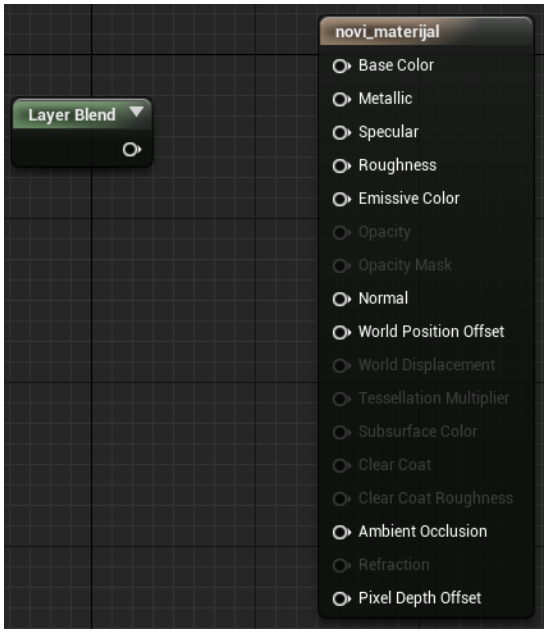
Slika 6.6 Izgled tla

Prvi korak je izrada novog materijala, materijal se izrađuje desnim klikom u mapu gdje se ga sprema te odabirom Material iz izbornika. Nakon što se materijalu da ime, materijal se može otvoriti. Početni materijal nema teksture ni nikakvih početnih opcija, te sve što je potrebno treba ručno napraviti.

6.4 Uređivanje novog materijala

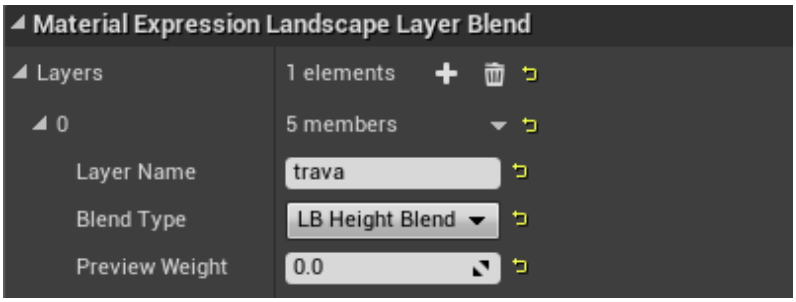
Novi prozor koji se otvori naziva se Material editor, on radi na sličnom principu kao i Blueprints skriptiranje tj. spajanjem blokova dobivaju seželjeni efekti.

Desnim klikom na praznu površinu otvorit će se izbornik u kojeg se može upisati što se traži. Prvo što je potrebno je Landscape layer blend.



Slika 6.7 Izgled blokova

Layer blend omogućava da se izmiješaju različite teksture zajedno u jednu. U njega se spajaju sve teksture koje se umeću u materijal te spajaju u pripadajuće točke na desnoj strani.

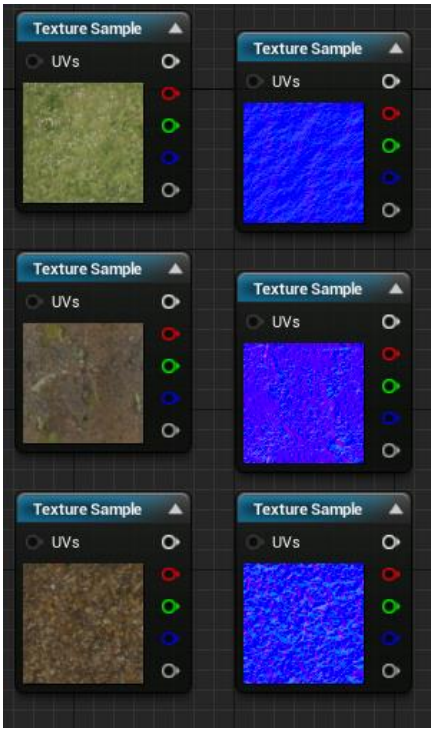


Slika 6.8 Opcije bloka Layer Blend

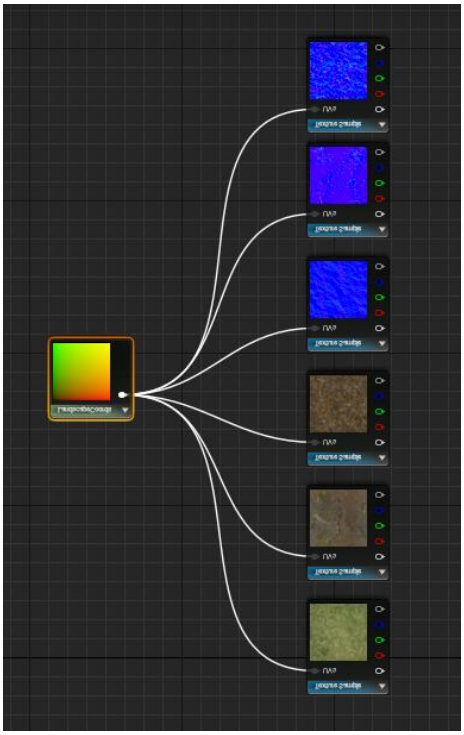
Sljedeći korak je dodavanja imena novih tekstura pritiskom na Layer blend u editoru te pritiskom na simbol „+“ u lijevom izborniku, on će napraviti novi materijal kojemu se onda da željeno ime, u ovom slučaju „trava“ te mijenjanje Blend tipa u LB_Height_Blend što omogućuje korištenje normal mapa u teksturi. Ovo se radi dok se ne dodaju sve željene teksture u naš mikser (eng. Blender).

6.5 Dodavanje tekstura

Dodavanje tekstura u materijal radi se klikom i povlačenjem u prozor editora. Teksture korištene ovdje nalaze se u open world demo paketu prethodno dodanom u projekt pod nazivom grass01, leafpath te forrestpath. Osim dodavanje samih tekstura potrebno je dodati i njihove normal mape koje se nalaze kraj njih u istoj datoteci. Prije nego se teksture spoje potrebno je dodati još jedan element pod nazivom Landscape Coordinates. On omogućava da mijenjamo veličinu naših tekstura u svijetu.

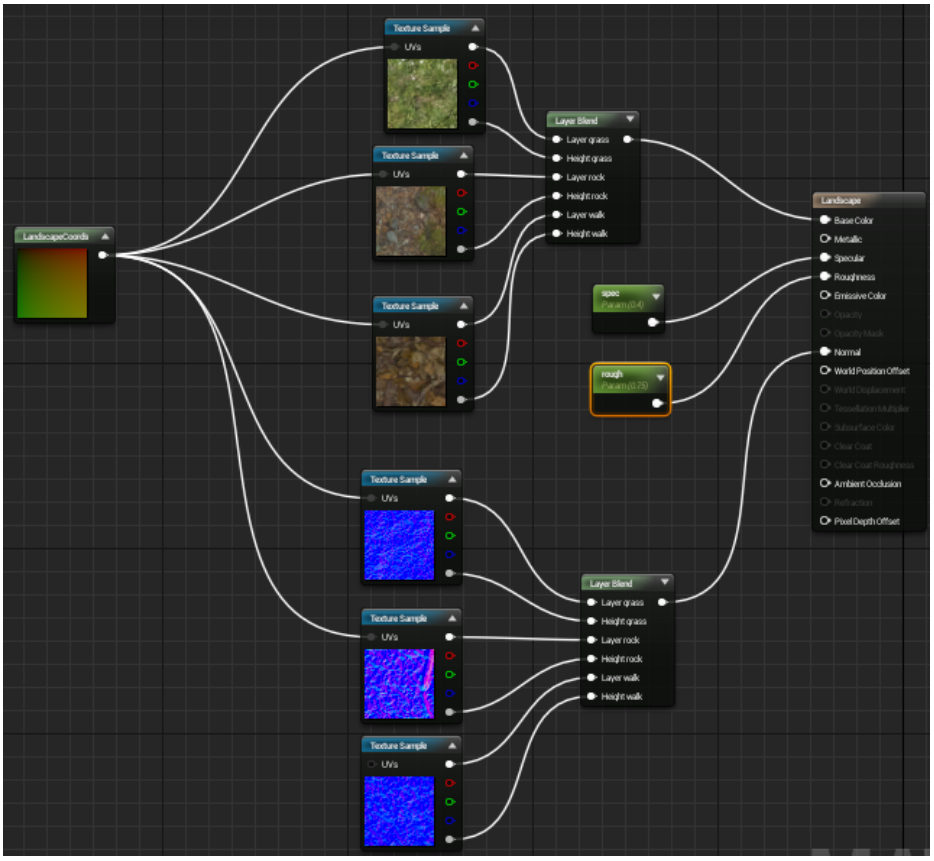


Slika 6.9 Teksture



Slika 6.10 Spajanje bloka

Nakon spajanja tekstura i koordinata spajaju se teksture u mikser, važno je da se mikser duplicira te se na isti način spoje i normal mape inače materijal neće biti valjani. Također za korištenje opcija refleksije i grubоće dodaje se novi element Scalar parameter koji se spoji u pripadajuće točkice na materijalu te jednostavnim mijenjanjem vrijednosti tog parametra određivati refleksiju i grubost materijala. Materijal nakon toga treba dodati kao materijal na element landscape.

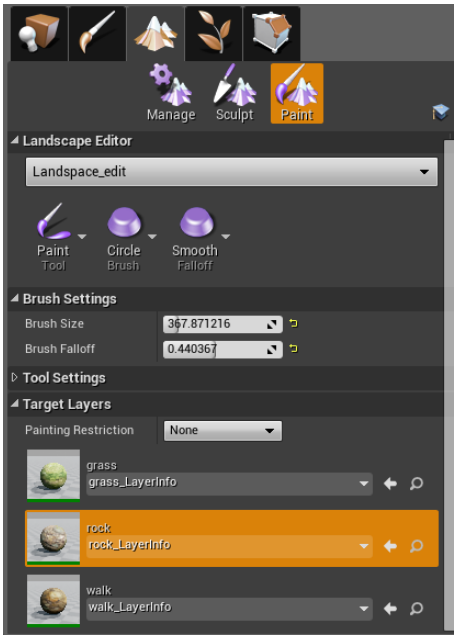


Slika 6.11 Konačni izgled materijala.

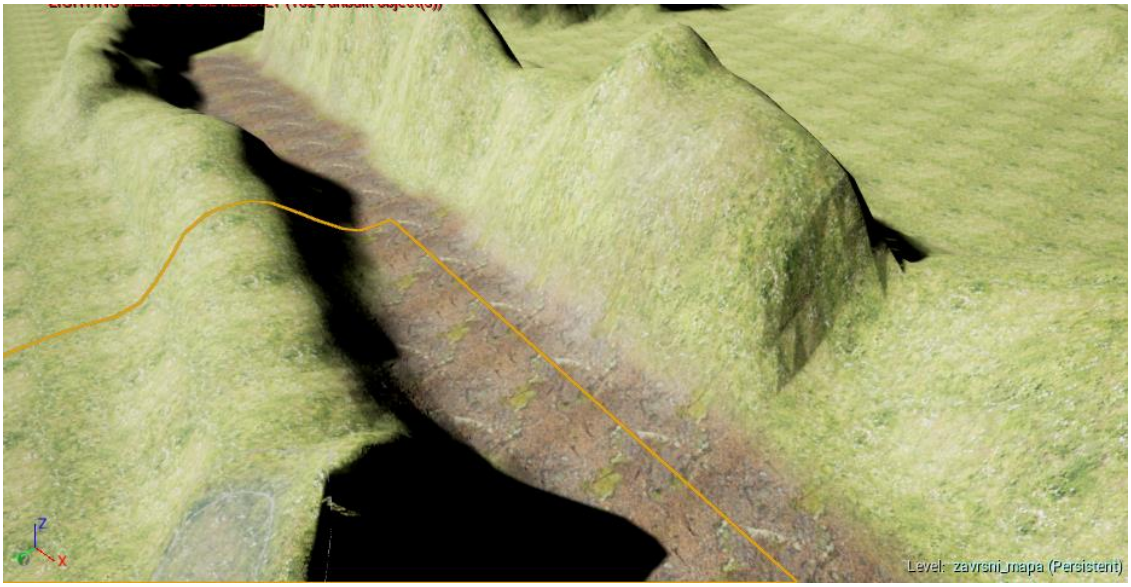
Slika 6.12 Izgled materijala u prozoru material editora

6.6 Bojanje mape novim materijalom

Za teksturiranje tla novim materijalom potrebno je prvo otići u pripadajući izbornik s opcijom bojanja gdje se biraju teksture te se jednostavno kistom dodaju na mapu. Kod prvog pokušaja bojanja biti će potrebno kliknuti na „+“ kraj imena teksture kako bi softver mogao napraviti novi sloj koji njemu omogućuje miješanje tekstura.



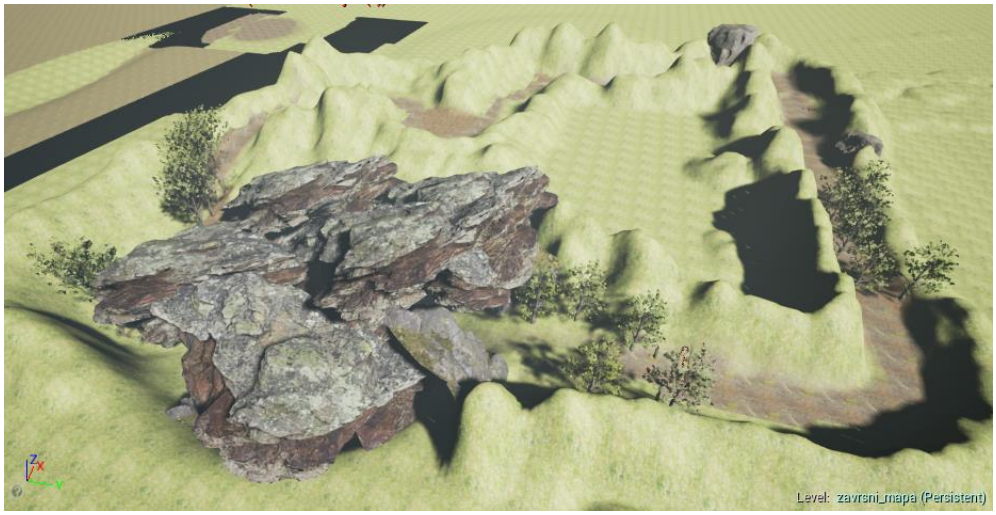
Slika 6.13 Sučelje za bojanje tekstura



Slika 6.14 Bojanje mape novim teksturama

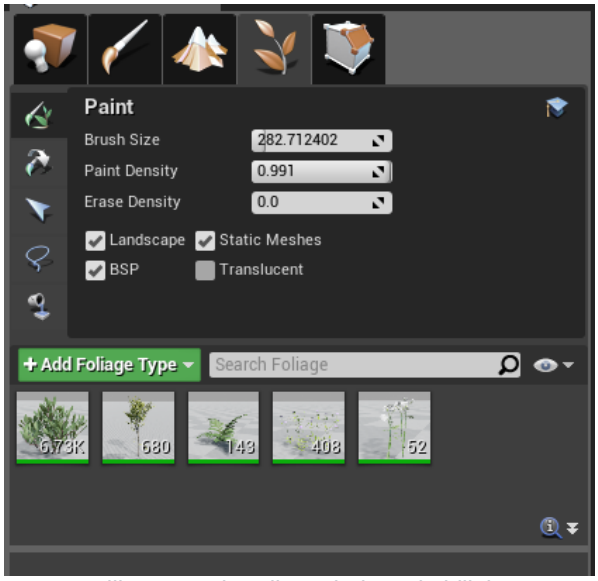
6.7 Dodavanje drveća i ostalih modela mape

Nakon bojanja mape potrebno je dodati drveće te još neke modele kako bi mapa bila što realističnija. Modeli se dodaju klikom i povlačenjem u scenu, na taj način moguće je manipulirati veličinom modela, rotacijom i naravno pozicijom, ova tehnika je korištena za dodavanje drveća duž stazu gdje se vozi igrač te za pećinu.



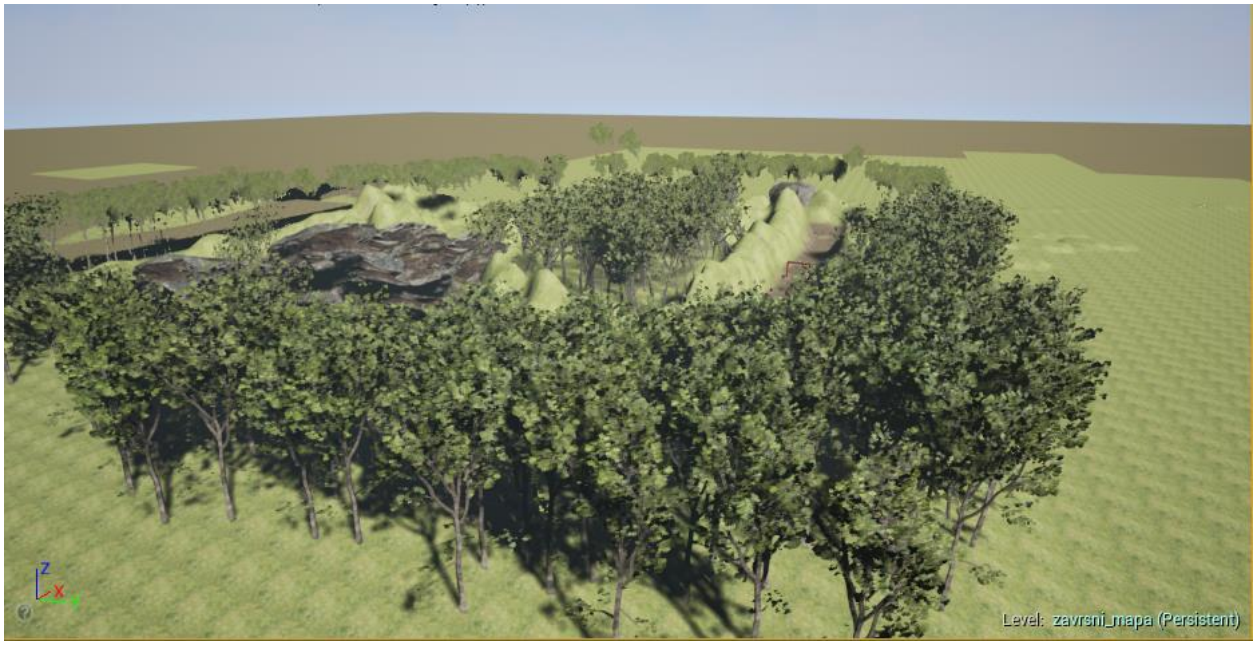
Slika 6.15 Izgled mape

Za drveće van mape korištena je opcija Foliage koja dodaje modele drveća u kist tako da se modeli dodaju „bojanjem“ po samoj mapi.



Slika 6.16 Izbornik za dodavanje biljaka

Modele se dodaje povlačenjem u izbornik ili korištenjem tražilice, a bojanje se vrši na isti način kao i kod bojanje tekstura. Ova tehnika stvara drveće kroz koje igrač prolazi, zbog toga nije korištena za drveće na stazi, pošto u slučaju sudaranja sa njima nije poželjno da igrač prođe kroz njih.



Slika 6.17 Konačni izgled mape iz zraka

Slika 6.18 Konačni izgled mape iz igračeve perspektive

7. Dodavanje igrača i priprema za programiranje

Prije početka samog programiranja potrebno je u mapu povući nekoliko dodatnih komponenti nužnih za samo programiranje. Pošto je programiranje rađeno unutar internog alata za skriptiranje, već prije spomenutog Blueprint alata, potrebno je napraviti nekoliko klasa u kojima će se odrađivati samo skriptiranje. Kod izrade klasa potrebno je odabrati roditelj klasu (eng. Parent class) od koje će klasa uzeti neka svojstva. Roditelj klase mogu biti:

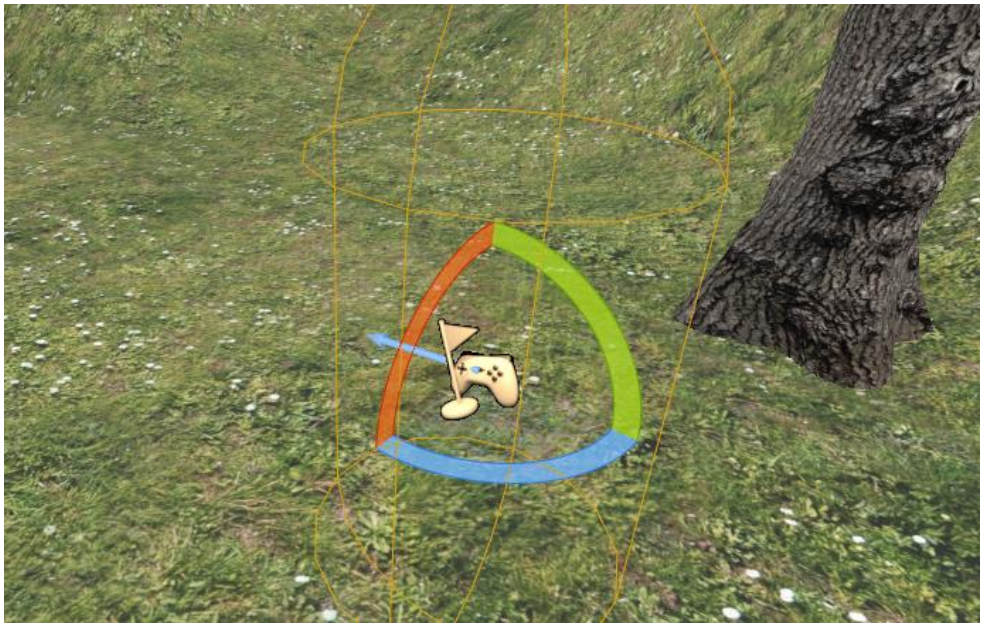
- Actor (objekt koji se može postaviti bilo gdje u svijetu)
- Pawn (objekt kojeg se može dobivati inpute od kontrolera)
- Character (objekt koji ima sposobnosti da hoda, skače i slično)
- PlayerController (objekt koji je odgovoran za kontroliranje igračevog lika)
- Game Mode (Game Mode definira igru koja se trenutno igra, njena pravila i slično)

Također osim gore navedenih klasa postoji i još mnoštvo drugih klasa s različitim namjenama. Sama izrada klasa je vrlo jednostavna.

Potrebno je desnim klikom miša kliknuti u mapu sa sadržajem te odabrati Blueprint class, nakon toga pojavit će se prozor koji će imati sve navedene roditelj klase. Za ovu igru korištene su roditelj klase Actor, PlayerController, SaveGame te posebna vrsta klase macro library koja je korištena da vrijeme pretvori u tekst koji je standardno zapisan u decimalnom zapisu u normalni zapis koji igra može prikazati na ekranu. Ostale klase služe za računanje vremena, spremanja vremena kako se može utvrditi najbolje vrijeme te brojanje krugova i provjeravanje da li je igrač prošao kroz kontrolne točke.

7.1 Dodavanje početne pozicije

Prije izrade samih klasa potrebno je uzeti jedan osnovni element sa lijeve strane izbornika pod imenom Player Start, kao što mu i samo ime kaže taj element označava gdje će igrač započeti igru tj. gdje će se njegov lik stvoriti kada igra počne. Element je zatim dovoljno samo povući u mapu na dio staze gdje za to najbolje odgovara, također za preciznost je moguće pritisnuti tipku End na tipkovnici koja će automatski spustiti element na zemlju. Također nužno je obratiti pažnju na plavu strelicu koja se vidi sa jedne strane elementa, ona upućuje na smjer u koji će igrač gledati. [10]



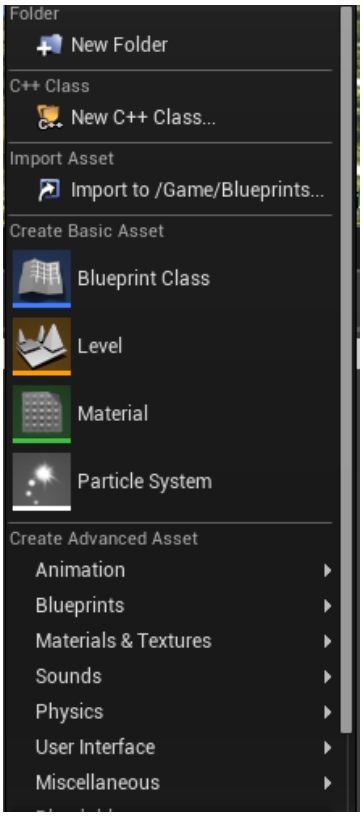
Slika 7.1 Izgled početne pozicije

Zahvaljujući ovom elementu moguće je u bilo kojem trenutku promijeniti gdje će igrač započeti igru povlačenjem elementa na drugo mjesto na stazi ili cijeloj mapi. Za testiranje je moguće pritisnuti Play na vrhu sučelja ili pritiskom ALT+P, igra će započeti na mjestu gdje je stavljen start element.

7.2 Izrada prvih klasa

Kod kreiranja klasa prva klasa koju je potrebno izraditi je klasa kontrolnih točka (eng. Checkpoint). Kontrolna točka služi za vraćanje igrača na njenu poziciju ukoliko se prevrne ili želi vratiti zbog bilo kojeg razloga, u drugim igrama ona može služiti kao točka na koju se igrač vraća nakon što umre ili izgubi. Za njenu izradu dovoljno je kliknuti desnim klikom u mapu te odabirom blueprint class a nakon toga Actor. Nakon što je klasa izrađena potrebno ju je nazvati, za ovaj projekt korišten je naziv checkpoints. [11]

Slika 7.2 Izbornik biranja roditelj klase



Slika 7.3 Izbornik izrade blueprinta

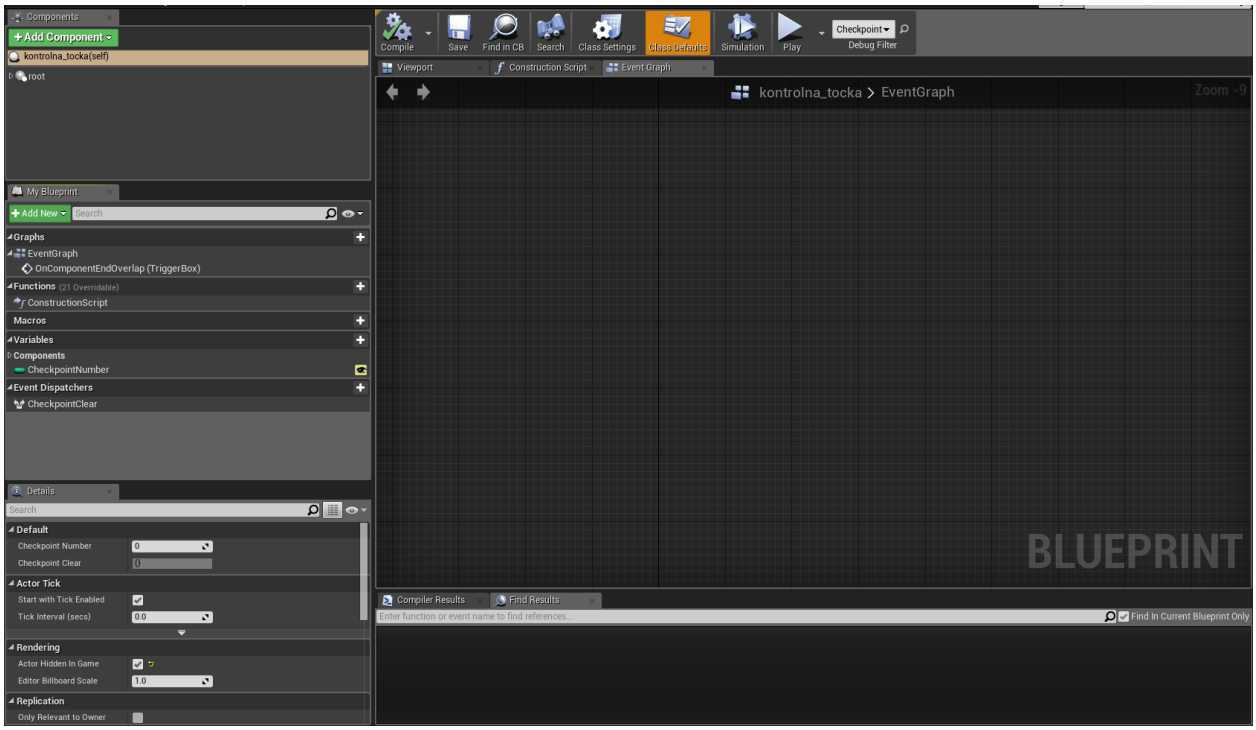
Istim postupkom izrađene su i ostale klase. Za klasu kontroler izabrana je roditelj klasa PlayerControler, za klasu SaveGame odabrana je roditelj klasa Save Game koristeći tražilicu vidljivu na slici gore i na kraju klasa koja će imati opcije utrke nazvana opcije za koju je odabrana roditelj klasa Actor.

8. Postavljanje sustava kontrolnih točaka

Za izradu sustava kontrolnih točaka nužno je prvo odrediti i napraviti nekoliko nužnih koraka kako bi cijeli sustav funkcionirao pravilno. Potrebno je odrediti treba li igrač vidjeti kontrolne točke ili ne te kako odrediti da li je igrač prošao kroz iste. Za svrhe testiranja za vrijeme izrade definitivno je lakše kada su točke vidljive i imaju neki fizički oblik kroz koji igrač mora proći. Što se tiče prolaska kroz iste to je najbolje odraditi detektirajući igračevu brzinu dok prolazi kroz kontrolnu točku umjesto njegove orijentacije. Ukoliko se detektira orijentacija, igra neće detektirati igrača da je prošao kroz točku ukoliko prolazi kroz točku unatrag, što je nepraktično pošto se igrač može preokrenuti taman prije ulaska u točku. Zbog toga je najbolje detektirati njegovu brzinu, ako prođe kroz točku sa bilo kojom brzinom računati će se kao korektan prelazak kroz točku i neće kažnjavati igrača ukoliko preokrene taman prije ulaska u kontrolnu točku.[20]

Što se tiče vizualne reprezentacije točaka, najbolje je koristeći sustave čestica ili neke 3D modele. Za svrhe testiranja poslužiti će i osnovni izgled kontrolnih točaka, a kasnije ga je uvijek moguće promijeniti kada uvidimo da sustav radi pravilno. Pošto je promjena vizualnog dijela ovog sustava najjednostavniji dio koji je moguće riješiti u par klikova najbolje se fokusirati na skriptiranje sustava kontrolnih točaka, a tek na kraju ulaziti u vizualni dio sustava kontrolnih točaka.

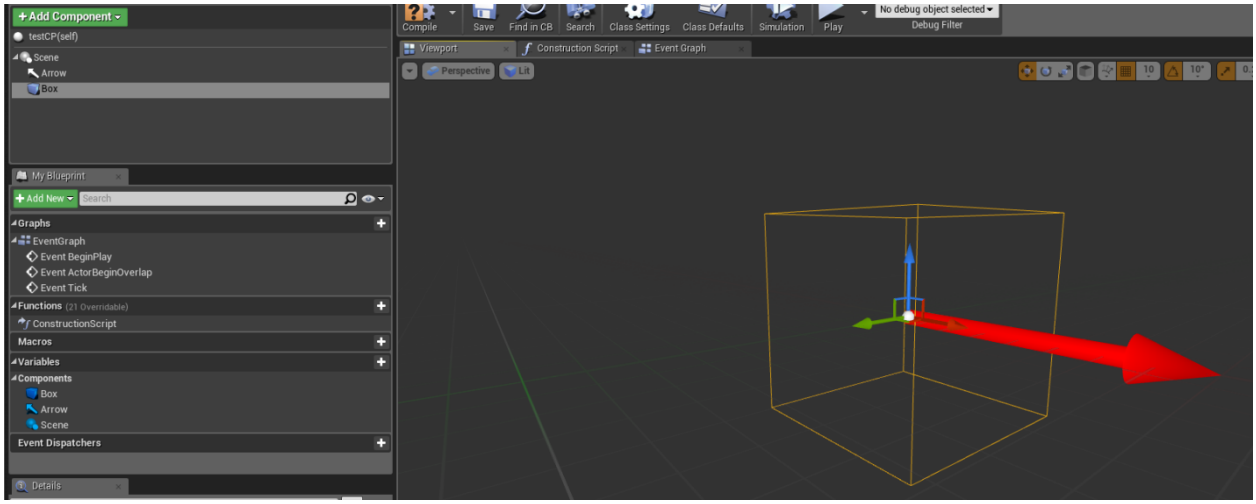
Prvi korak kod kretanja izrade ovog sustava je otvaranje klase iz mape. Ono što će se otvoriti je blueprint editor. [12]



Slika 8.1 Izgled sučelja blueprinta

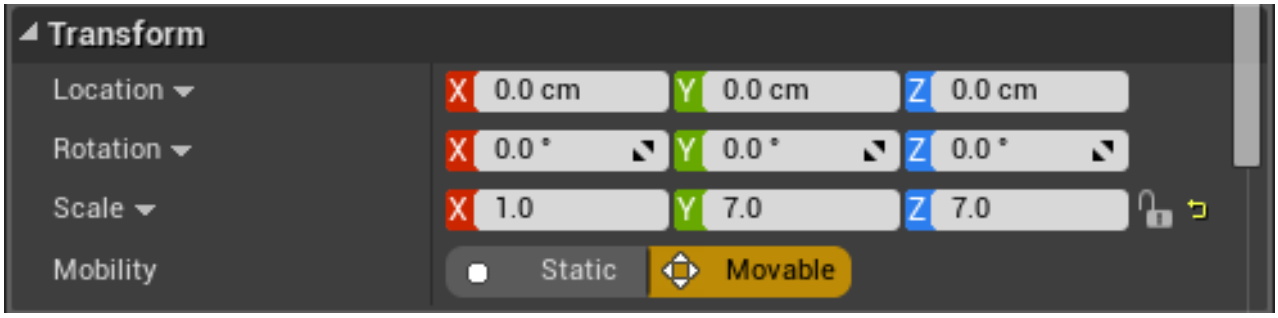
U ovome prozoru su sve opcije koje su nužne za izradu ovoga sustava. U sredini se nalazi ploča za naše nacrte (eng. Blueprints) a iznad nje imamo 3 prozora između kojim se možemo izmjenjivati, Viewport – koji nam daje 3D prikaz kako ono na čemu se radi izgleda ukoliko je to moguće, Constuction Script te Event Graph. U ovom slučaju svo skriptiranje radi se unutar Event graph prozora, to je prozor u kojem se programiraju događaji vezani uz kontrolne točke.

Prvi korak je da se otvori Viewport za 3D prikaz onoga na čemu se radi. Nakon toga potrebno je napraviti novu scenu klikom na Add component tipku u gornjem lijevom kutu te upisom ključne riječi scene u tražilicu te odabrati element „Scene“ koji će se pojaviti. Nakon toga moguće je element preimenovati ili ostaviti pod nazivom scene te ga klikom povući na Default Scene Root element koji se nalazi iznad njega kako bi ga zamijenili.Zatim se dodaju elementi Box Collision, Arrow i Particle System.



Slika 8.2 Sučelje blueprinta

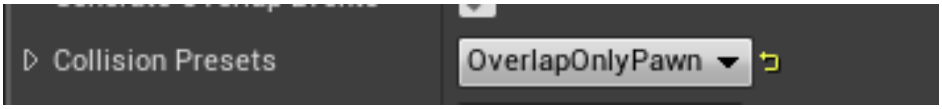
Box će služiti kao vizualna reprezentacija prostora kroz koji igrač mora proći kako bi aktivirao kontrolnu točku dok strelica služi kao indikator smjera u kojem mora proći kroz kontrolnu točku. Sada je potrebno povećati Box pošto je sada premali da bi igrač prošao kroz njega, a to se može selektirajući Box te promjenom Y i Z veličine na 7.



Slika 8.3 Postavke boxa

Osim veličine potrebno je još promijeniti 2 opcije, a to su HIdden ingame kojemu je potrebno maknuti kvačicu te Collision Presets na OverlapOnlyPawn. Prva opcija omogućava da se Box vidi u igri dok druga opcija označava da samo Igrač može proći i aktivirati kontrolnu točku, to sprečava aktivaciju točaka iz drugih izvora koji nisu igrači.

Slika 8.4 Opcija sakrivanja u igri



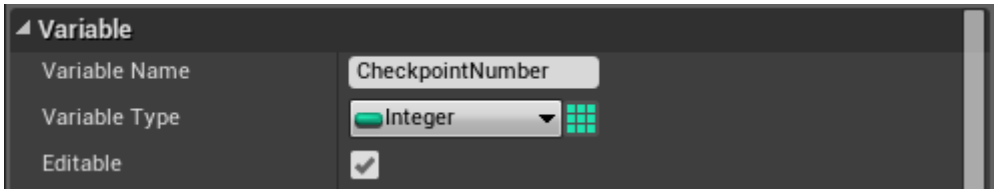
Slika 8.5 Opcija preklapanja elemenata

Prije početka skriptiranja potrebno je napraviti varijable koje će klasa koristiti. U slučaju kontrolnih točaka potrebna je jedna varijabla koja će označavati koliko kontrolnih točaka ima trenutno u stazi te također i takozvani Event Dispatcher koji će ostalim klasama javiti da je kontrolna točka bila aktivirana. Pošto će postavke sa vremenima, brojem krugova i broj kontrolnih točaka kontrolirati druga klasa koja će ovoj klasi pošiljati vrijednosti ovdje neće biti definiran broj kontrolnih točaka. [13]

Za izradu varijable potrebno je pritisnuti na plus kraj varijabli te nazvati varijablu.

Slika 8.6 Komponente

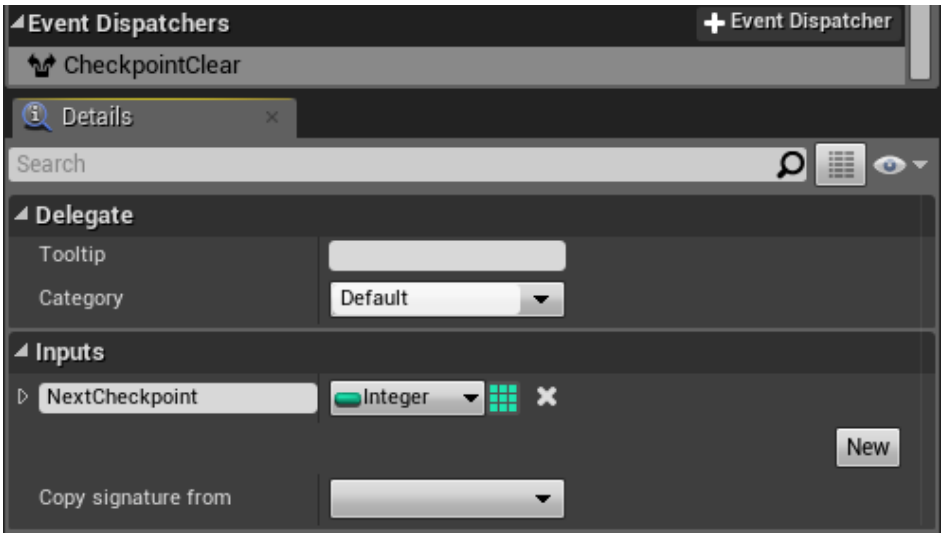
Nakon izrade varijable potrebno je promijeniti njen tip u integer te staviti kvačicu na Editable.



Slika 8.7 Postavke varijable

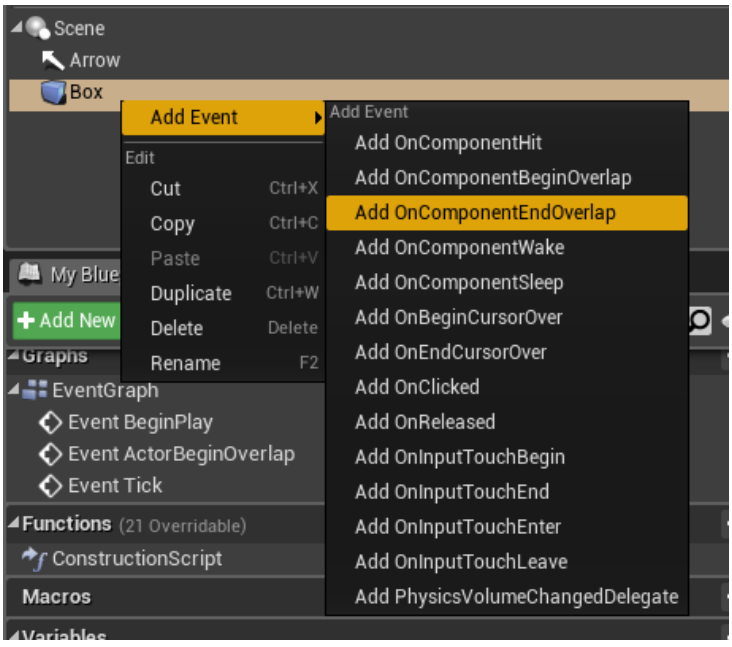
Na isti način potrebno je izraditi novi dispatcher koji će ostalim klasama davati do znanja da li je kontolna točka stvarno aktivirana.

Sada je potrebno pritisnuti New pod opcijom Inputs te napraviti novi input koji će prenijeti koja kontrolna točka treba biti sljedeća nakon što je trenutna aktivirana.



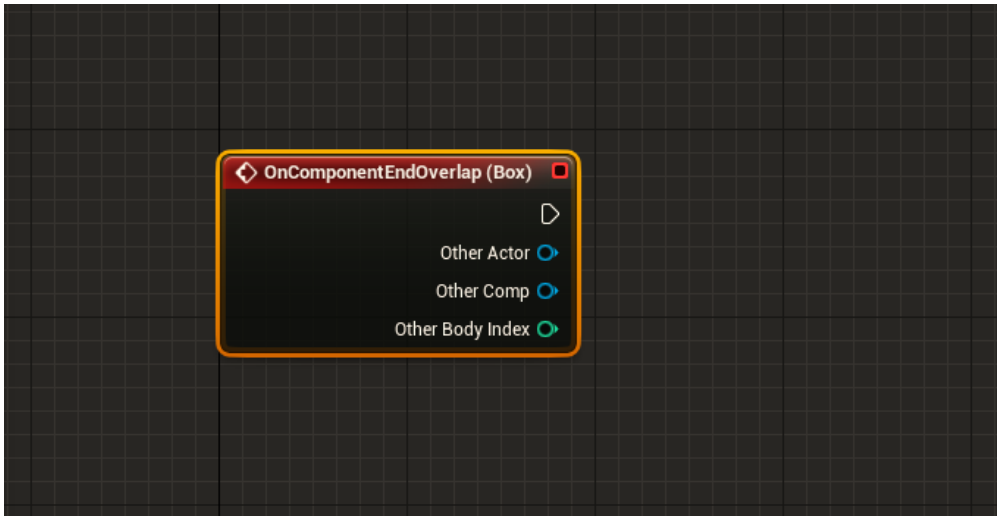
Slika 8.8 Izrada dispatchera

Sljedeći korak je izrada skripte koja detektira prolazak igrača kroz prije izrađeni Box. Za početak potrebno je pritisnuti desnim klikom Box, te odabrati opciju Add Event, te OnEventOverlap. Taj događaj označava da će kontolna točka biti prijeđena samo kada igrač kroz nju prijeđe, a ne kada ju dodirne, zato je u imenu End a ne Begin.



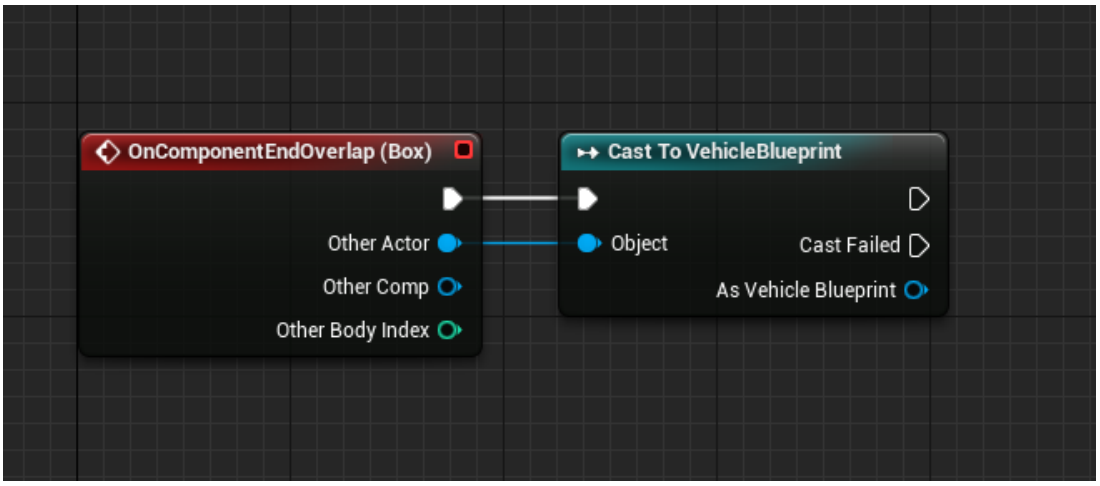
Slika 8.9 Dodavanje događaja

Prozor će se automatski prebaciti u Event Graph u kojemu je sada dobro obrisati sve osim novo napravljenog događaja.[7]



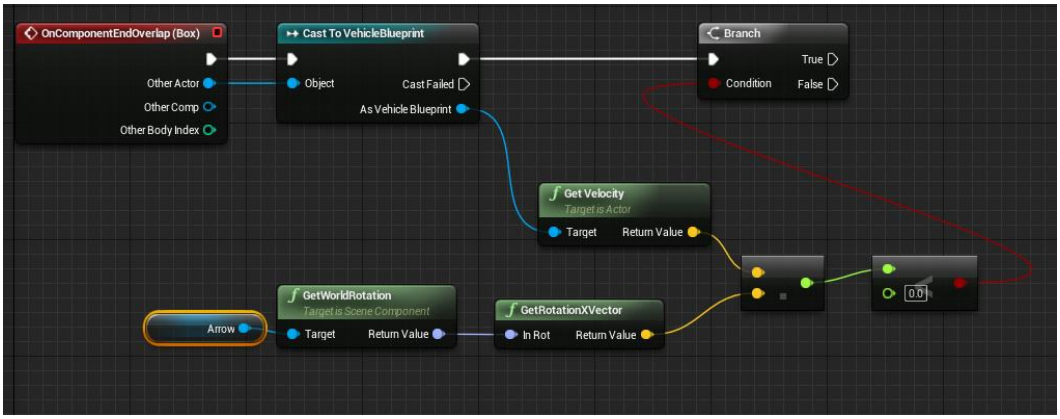
Slika 8.10 Izgled napravljenog događaja

Pritiskom na krug kraj Other Actor te povlačenjem u prazni prostor otvara se prozor tražilice kroz koji se dodaju sve klase i varijable koje su moguće, u ovom slučaju potrebno je dodati element Cast To VehicleBlueprint. Time se daje do znanja da se ovaj događaj bazira za vozilo koje će igrač voziti, tj. da će vozilo biti to koje će se provjeravati za uspješnost prolaska kroz kontrolnu točku. Ukoliko nije vozilo to koje je prošlo kroz kontrolnu točku neće se dogoditi ništa.



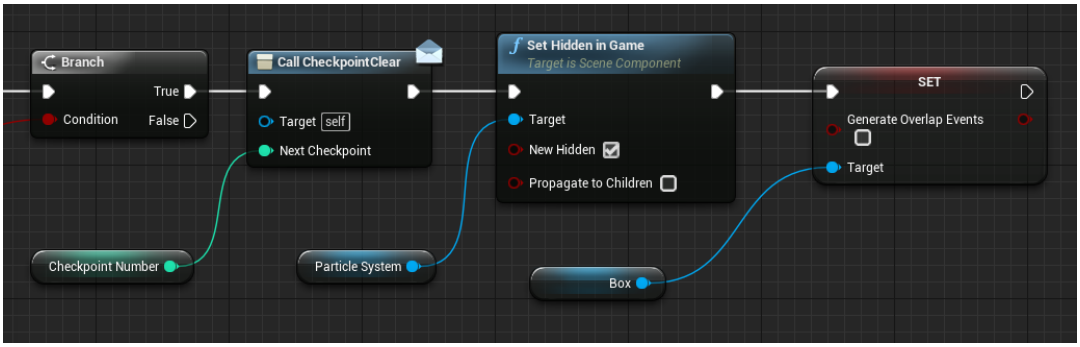
Slika 8.11 Spajanje blokova

Ono što je nužno napraviti je detektirati prolazak vozila kroz Box. To je ostvareno tako da se prvo povuče prije izrađena strelica i ispusti u skriptu. Pošto je već prije objašnjeno da ona služi kao indikator smjera u kojemu igrač mora proći, njen smjer se može kombinirati zajedno sa brzinom vozila. Koristeći Branch, element koji uzima uvjet te zatim ovisno o istinitosti uvjeta radi određenu radnju, ispitano je da li je vozilo prošlo kroz točku u pravom smjeru.



Slika 8.12 Izgled prvog dijela skripte

Ukoliko je uvjet pogrešan neće se ništa desiti, no ukoliko je uvjet točan pozvat će se event dispatcher koji daje ostalim klasama i funkcijama do znanja da je kontrolna točka prijeđena. Također se skrivaju sustavi čestica i Box, to će značiti da prijeđena kontrolna točka postaje nevidljiva nakon prelaska kroz nju, a ona koja slijedi nakon nje postaje vidljiva. Box sam po sebi nije potrebno sakriti preko skripte već je najbolje ga sakriti u njegovim opcijama po završetku samog skriptiranja i programiranja svih ostalih elemenata. On služi samo kao vizualni prikaz kontrolnih točaka koji će se kasnije zamijeniti konkretnim izgledom kao što su čestice ili 3D model.

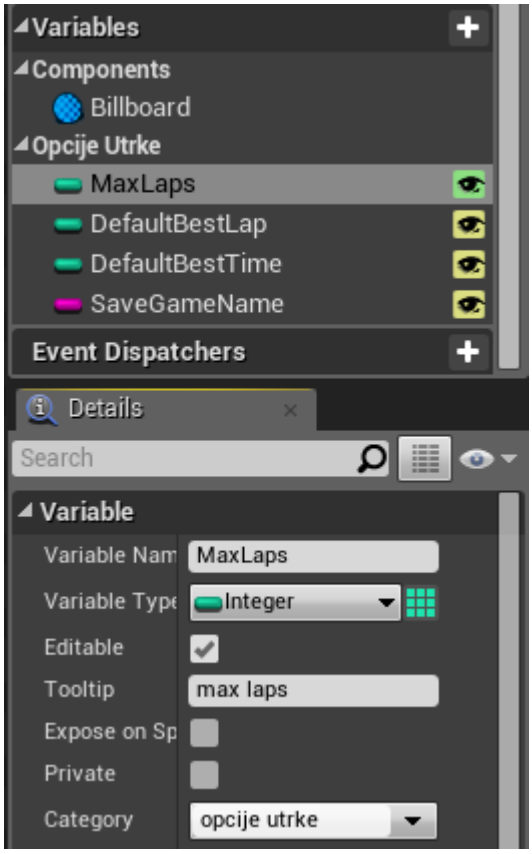


Slika 8.13 Izgled drugog dijela skripte

9. Početak programiranja sustava krugova

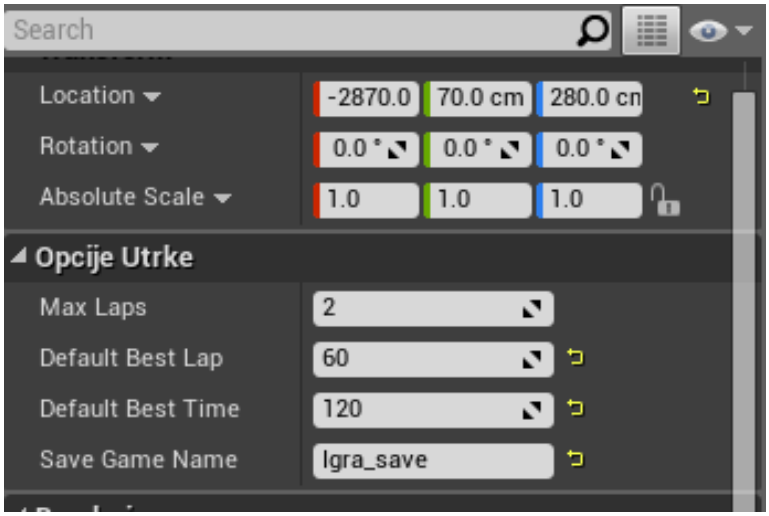
Sustav krugova predstavlja skriptu koja će brojati koliko krugova je odrađeno u utrci. Ukoliko su prijeđeni svi zadani krugovi igra završava, ukoliko nisu nastavlja se u slijedeći krug.Za ovaj sustav koristi se blueprint „opcije“ koji je prethodno izrađen. Unutar njega za 3D prikaz blueprinta može se koristiti komponenta Billboard koja nije ništa drugo nego 2D slika koja je uvijek okrenuta prema kameri, to nam omogućava da blueprint povućemo u samu mapu te ga uvijek selektiramo iz glavnog prozora igre. Time je vrlo jednostavno promijeniti broj krugova i vremena utrke iz glavnog izbornika bez potrebe da se dodatno otvaraju blueprinti i traži se po skriptama.

Za početak potrebno je stvoriti nekoliko varijabli koje igra treba, a to su varijable za broj krugova, zadana vremena, vremena koja igrač postigne, broj kontrolnih točaka te još varijable za spremanje igre.



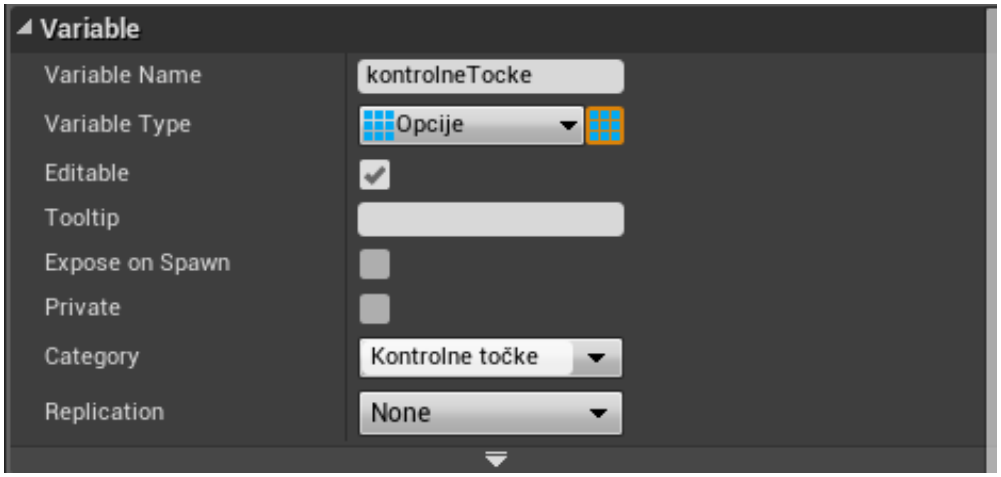
Slika 9.1 Izbornik postavki varijabli

Varijable broja krugova te vremena su integeri dok je ime za spremanje igre String. Ono što je bitno kodizrade ovih varijabli je stavljanje opcije „Editable“ na istinito, tj. stavljanje kvačice kraj nje. To označava da vrijednost varijable nije fiksna vrijednost već se može uvijek mijenjati po želji. Po izradi varijabli pritiskom na tipku compile i save spremaju se sve promjene. To znači da je element moguće dodati u samu igru jednostavnim povlačenjem. Selektiranjem elementa vidi se da su sve opcije gdje trebaju biti. Opcije je zatim moguće urediti po želji.



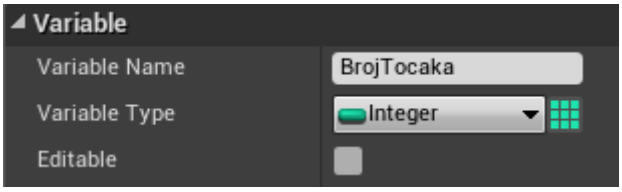
Slika 9.2 Izbornik sa napravljenim opcijama

Sljedeća varijabla je za kontrolne točke. Nju je dovoljno nazvati kontrolneTocke, a kategoriju pod koju spada Kontrolne točke. Ono po čemu se ova varijabla razlikuje od ostalih je da njen tip nije integer već „opcije“ tj. njen tip je ime Blueprinta u kojemu se kontrolne točke stvaraju (ranije izrađeni blueprint). Varijablu je potrebno pretvoriti u listu (eng. Array)[14] pritiskom na ikonu kraj opcije tipa varijable. Razlog toga je spremanje svih kontrolnih točaka u ovu varijablu. Sa svim kontrolnim točkama u jednoj varijabli, korištenjem jednostavne petlje može se riješiti problem stvaranja sljedeće kontrolne točke kada je prethodna prijeđena. Pošto petlja ide od prvog elementa u listi do zadnjeg, sve što je potrebno je dodati sve kontrolne točke po redu kako su raspoređene u listu.



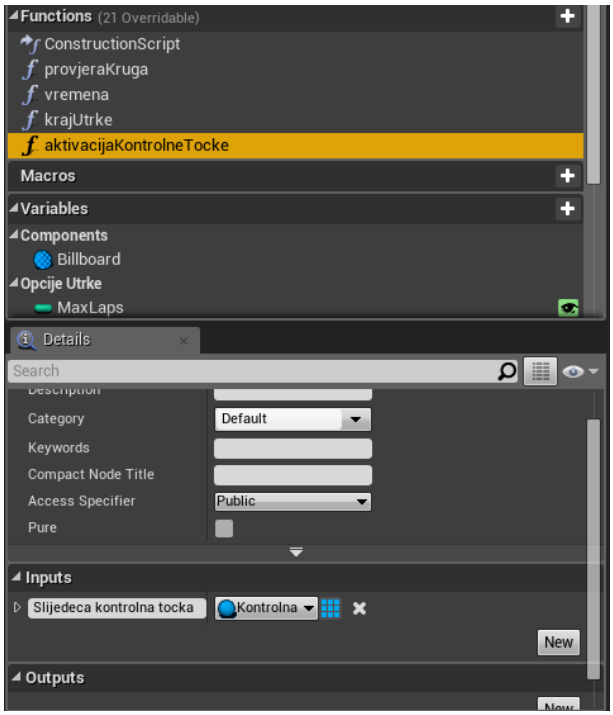
Slika 9.3 Postavke liste

Zadnja varijabla potrebna je broj kontrolnih točaka, ta varijabla će biti tipa integer bez kategorije i bez opcije Editable.



Slika 9.4 Varijabla točaka

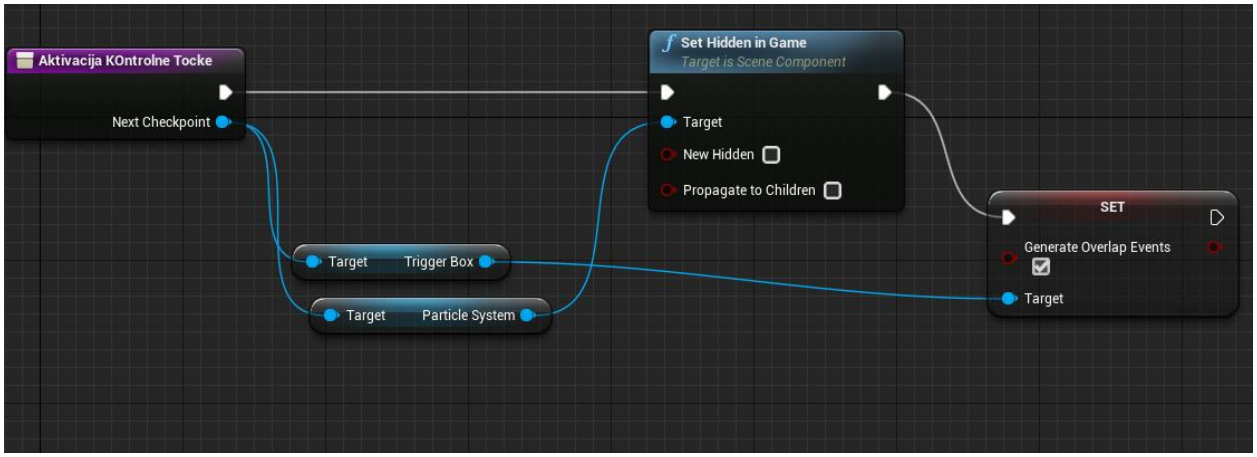
Nakon nje potrebo je stvoriti funkcije koje će ovaj blueprint obavljati. Te funkcije su provjera ako je utrka završila, provjera završetka kruga, mijenjanje rekordnog vremena sa igračevim vremenom ukoliko postigne bolje vrijeme te aktivacija slijedeće kontrolne točke nakon što je prva prijeđena. Kod stvaranja funkcija važno je odrediti koje vrijednosti funkcija prima te koje i kada vrijednosti vraća. Kod funkcije koja provjerava završetak utrke nije potrebno primati inpute ali je bitno da ona vraća vrijednost točno ili netočno ovisno o završetku utrke. Izrada funkcija radi na istom principu kao i izrada varijabla, a opcije o inputima i outputima nalaze se na dnu opcija kod izrada funkcije.



Slika 9.5 Lista funkcija

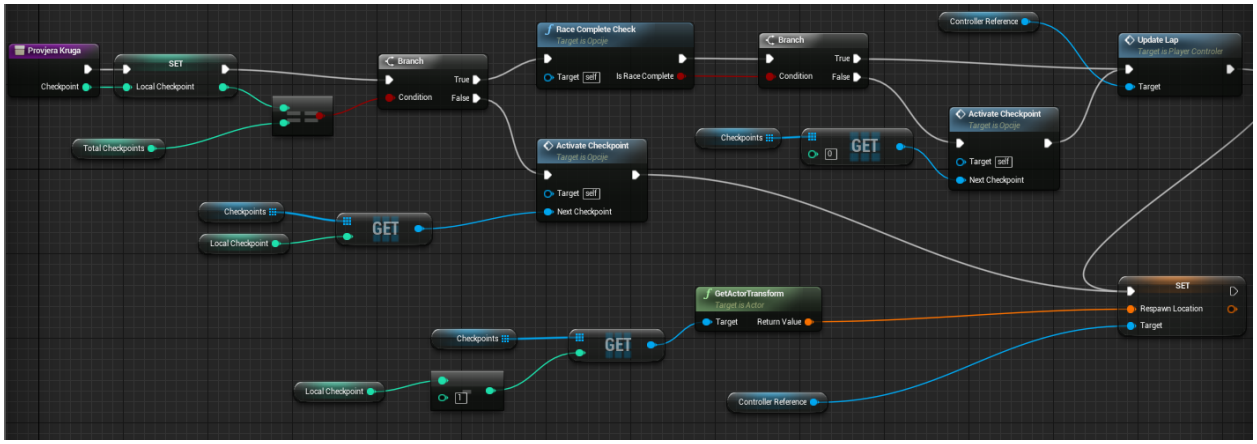
9.1 Pisanje skripti u funkcije

Nakon dodavanja funkcije kreće se na izradu skripti unutar funkcija. Izrada skripti radi na principu dodavanja blokova na platno te spajanja vrijednosti koje se nalaze na njima. Svaki blueprint ili funkcija unutar blueprinta uvijek ima početni blok u sebi sa nazivom blueprinta ili funkcije, na taj blok dodaju se svi ostali blokovi koji obavljaju neku radnju, time se prilikom pozivanja te funkcije izvršava skripta. Sami blokovi su funkcije koje postoje u samom Unreal Enginu, na primjer blok Posses koji daje igraču kontrolu nad objektom, blok Destroy koji uništava element u igri ili blok Set Hidden in Game koji skriva objekte u igri.



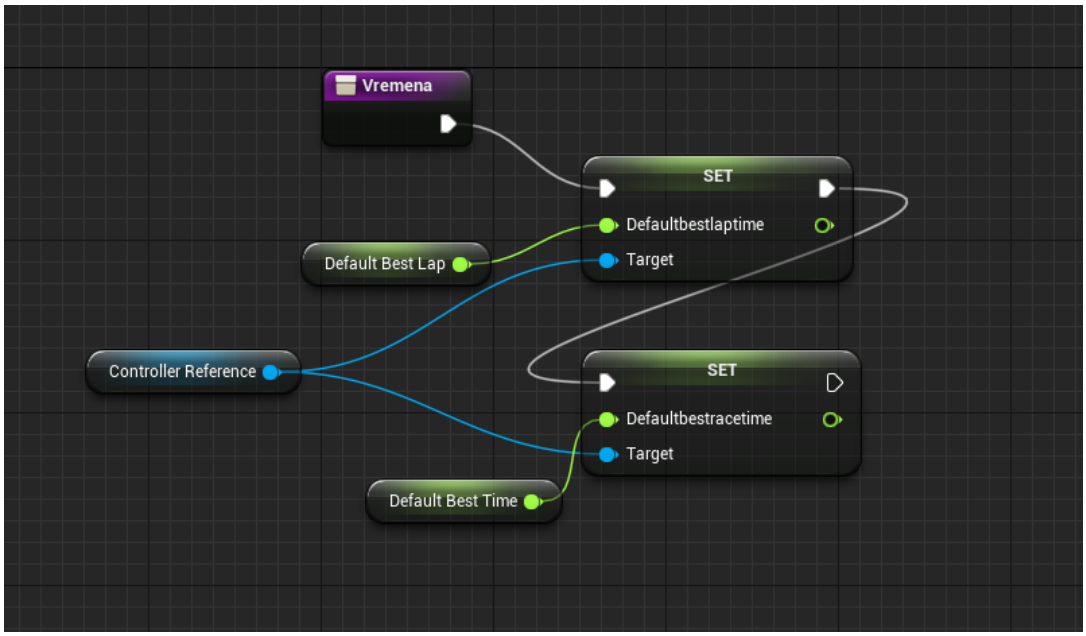
Slika 9.6 Skripta aktivacije kontrolnih točaka

Ova skripta sakriva particle system koji se pojavljuje kao kontrolna točka. Na početni blok dodan je blok Set Hidden in Game koji sakriva elemente u igri. Elemente koje sakriva potrebno je spojiti u vrijednost Target. Zadnji blok koji je dodan je Generate Overlap Event koji pokreće događaje vezane uz preklapanje objekta (preklapanja vozila i kontrolne točke).



Slika 9.7 Skripta za provjeru krugova

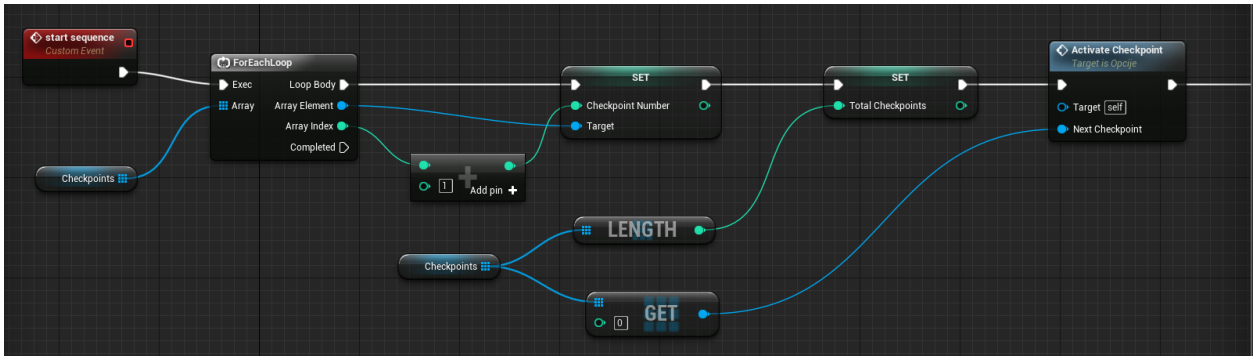
Ukoliko je više krugova u igri ova skripta provjerava da li je igrač prošao kroz sve krugove u igri, te da li je završio utrku prošavši kroz sve kontrolne točke. Ovdje korištenjem bloka Branch koji uzima uvjet i vraća vrijednost točno ili netočno se provjerava ako trenutna kontrolna točka je zadnja u nizu. Ako kontrolna točka nije zadnja u nizu poziva se blok Update Checkpoint koji stvara sljedeću, a ako kontrolna točka je zadnja u nizu poziva se blok Update lap koji pokreće sljedeći krug.



Slika 9.8 Skripta koja postavlja vremena

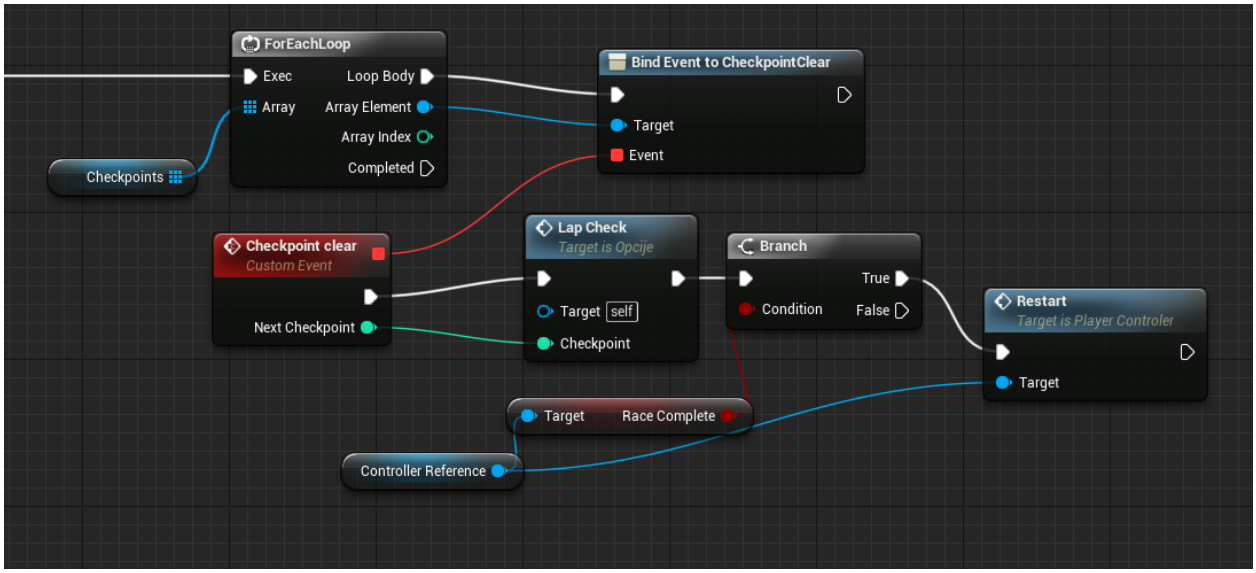
Skripta koja zadaje početna najbolja vremena za cijelu utrku te za jedan krug. Varijable Default Best Lap i Default Best time se spajaju na blok SET koji sprema njihove vrijednosti. Ovo su vrijednosti koje će biti vidljive igraću zbog toga je nužan ovaj postupak.

Nakon svih skripti u funkcijama potrebno je napraviti i nekoliko blokova skripta i u Graph Editoru. Graph editor je najlakše definirati kao glavnu funkciju jednog blueprinta. Prvo je potrebno pozvati kontroler i započeti igru. Također je potrebno i pozvati funkcije te događaje iz drugih blueprinta. Neki od događaja koji su potrebni nisu ni napravljeni te ih je potrebno napraviti u pravom blueprintu. Događaji poput ponovnog pokretanja igre nakon završetka utrke svi će biti spremljeni u kontroleru kao glavnom blueprintu za sve funkcije vezane uz pokretanje, spremanje i završavanje igre.



Slika 9.9 skripta sa petljom

Ovo je prvi dio skripte koja pokreće sustav kontrolnih točaka, postavlja prvu kontrolnu točku te ukoliko igrač prođe kroz kontrolnu točku sakriva prijašnju točku te stvara novu kontrolnu točku ispred njega. Skripta je odrađena na način da petlja aktivira prvu kontrolnu točku u listi, a nakon prelaska kroz nju skripta će ju sakriti, a sljedeća točka postatće vidljiva na mapi.



Slika 9.10 Drugi dio skripte

U drugom dijelu skripte nalazi se petlja koja je spojena na blok koji poziva događaj Checkpoint Clear. Blok Checkpoint clear te blok Bind event to Checkpoint clear se sami spoje nakon dodavanja samo jednoh od njih. Na blok Checkpoint clear je dodan blok Lap Check koji poziva funkciju Lap check i provjerava ako je završio zadnji krug. Kao uvjet koristi se boolean varjabla Race complete koja vraća vrijednosti true ako je zadnji krug završio. Vraćanjem vrijednosti true poziva se blok Restart koji ponovno pokreće igru.

9.2 Stvaranje varijabli u kontroleru

Prije nego je moguće završiti sve skripte u blueprintu „opcije“ potrebno je stvoriti nekoliko funkcija i varijabli u kontroleru. On ujedno ima i najviše skripti pošto se radi o skriptama koje pokreću i zaustavljaju sve ostale. Za kontroler se može reći da je centar tj. glavni blueprint u cijelom projektu. Potrebno je napraviti funkcije koje učitavaju i spremaju rezultate igre, inicijalizirati sav tekst te izraditi funkciju koja provjerava vremena kruga i cijele utrke te ih pravilno prikazuje na ekranu. Za to će trebati napraviti skriptu koja pretvara brojčane zapise u tekstualni oblik koji se može prikazati na ekranu. Unreal omogućuje pisanje makro naredbi koje se mogu pozvati kasnije da odrade neki zadatak. U ovom slučaju makro komande će biti korištene za pretvorbu zapisa vremena u tekstualni oblik. Vrijeme u Unrealu je prikazano samo u sekundama, a pošto igra treba imati minute, sekunde i milisekunde potrebno je pretvoriti vrijeme u takav zapis. Također za prikaz na ekranu potrebno je pretvoriti vrijeme u varijablu string kako bi se cijelo vrijeme moglo prikazati na ekranu kao jedna linija brojki.

Prvo je potrebno stvoriti nužne varijable. Varijable trenutnog kruga i maksimalnog broja krugova su tipa integer dok su sve ostale varijable tipa float.

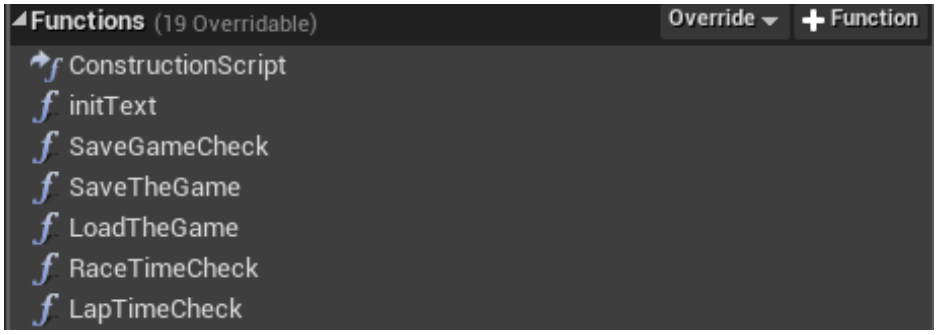
Slika 9.11 Varijable u kontroleru

Osim standardnih varijabli potrebne su i varijable tipa Text za vrijeme, krugove, ime mape te tekst koji će ispisivati odbrojavanje kod početka utrke.

Slika 9.12 Tekst varijable

Zadnje varijable koje su potrebne su boolean varijable za početak i kraj utrke. Boolean je tip varijable koji može imati vrijednost True ili False. Osim tih varijabla potrebna je varijabla respawnLocation tipa Transform. Ta varijabla koristit će se kod vraćanja igrača na prijašnju kontrolnu točku ukoliko se on prevrne i pritisne tipku koja će ga automatski vratiti na prijašnju kontrolnu točku. Zadnja varijabla je Save tipa Text koja u sebi nosi ime spremljene igre, u ovom slućaju to je G_SaveGame.

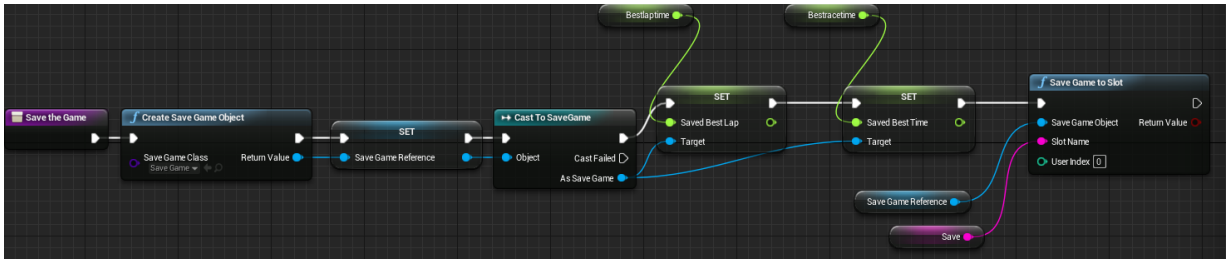
Slika 9.13 Boolean varijable



Slika 9.14 Funkcije

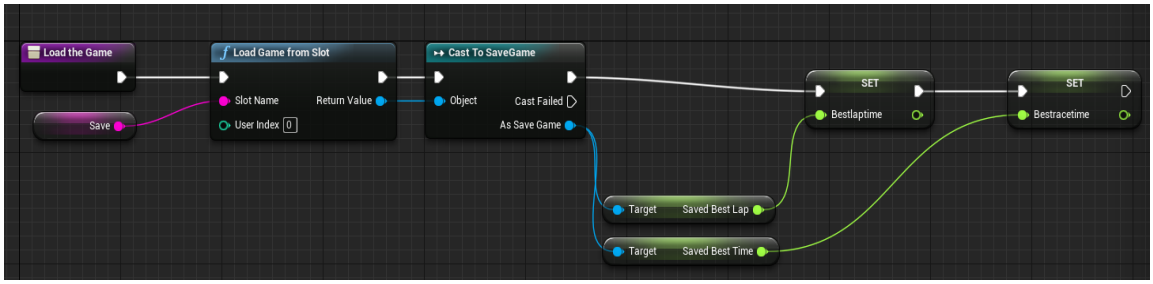
Kontroler ima 6 funkcija: pokretanje teksta, provjera ima li spremljene igre od prije, spremanje igre, učitavanje igre, te provjere vremena za krug i utrku. Osim njih potreban je i dispatcher pod nazivom UtrkaPocela koji će samo dati do znanja drugim funkcijama da je utrka stvarno počela.[15]

Prve 2 skripte koje su ujedno među najmanjima su Load i Save game. Njih je najbolje odraditi prvo pošto su vrlo jednostavne.



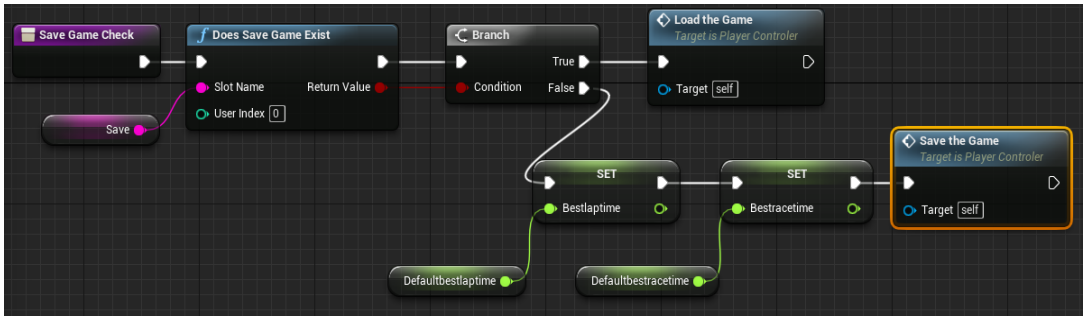
Slika 9.15 Skripta spremanja igre

Skripta za spremanje igre uzima vremena te ih stavlja u novi Save slot koji Unreal sam izrađuje u pozadini. Ovo je automatizirani postupak te jedina stvar koja je potrebna je pravilno spojiti elemente koje treba spremiti ostalo će se samo izvršiti.



Slika 9.16 Skripta učitavanaj igre

Učitavanje spremljene igre je još jednostavnije, skripta samo učita spremljenu igru te varijable koje su spremljene, u ovom slučaju najbolja vremena utrke te ih stavlja umjesto zadanih vremena.



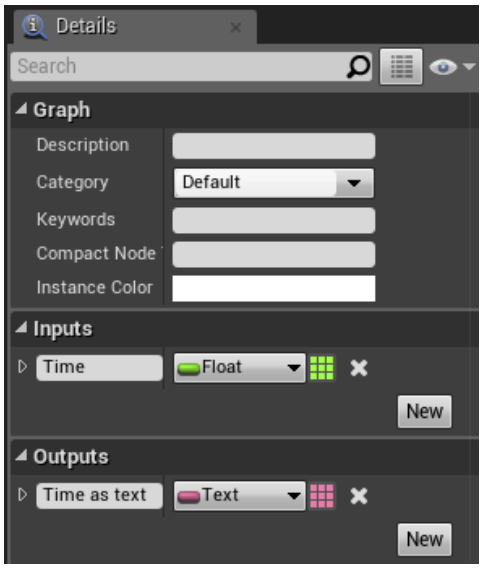
Slika 9.17 Skripta provjere spremljenih igara

Posljednja skripta provjerava postoji li spremljena igra, ukoliko postoji učitava ju, ukoliko ne postoji učitavaju se zadana početna vremena utrka te se spremaju.

Za pretvaranje vremena u tekstualni oblik korišten je makro sustav Unreal engina.

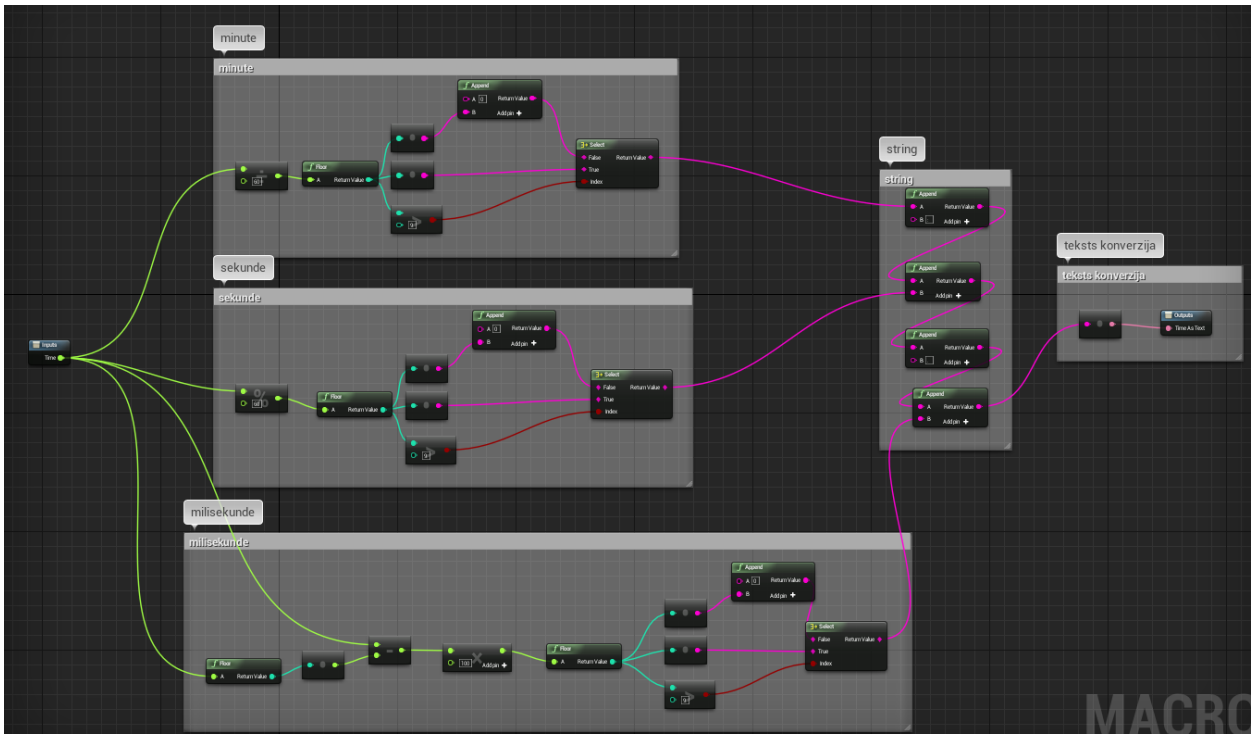
Blueprint makro je blok koji prima vrijednosti, obrađuje ih te vraća van nove vrijednosti. Najbolji primjer je makro koji prima vrijednosti 'a' i 'b', a vraća vrijednosti 'a+b'.

Kod prvog otvaranja makro blueprinta potrebno ga je nazvati, u ovom slučaju nazvan je time to text. Nakon preimenovanja treba zadati ime i tip inputu i outputu.

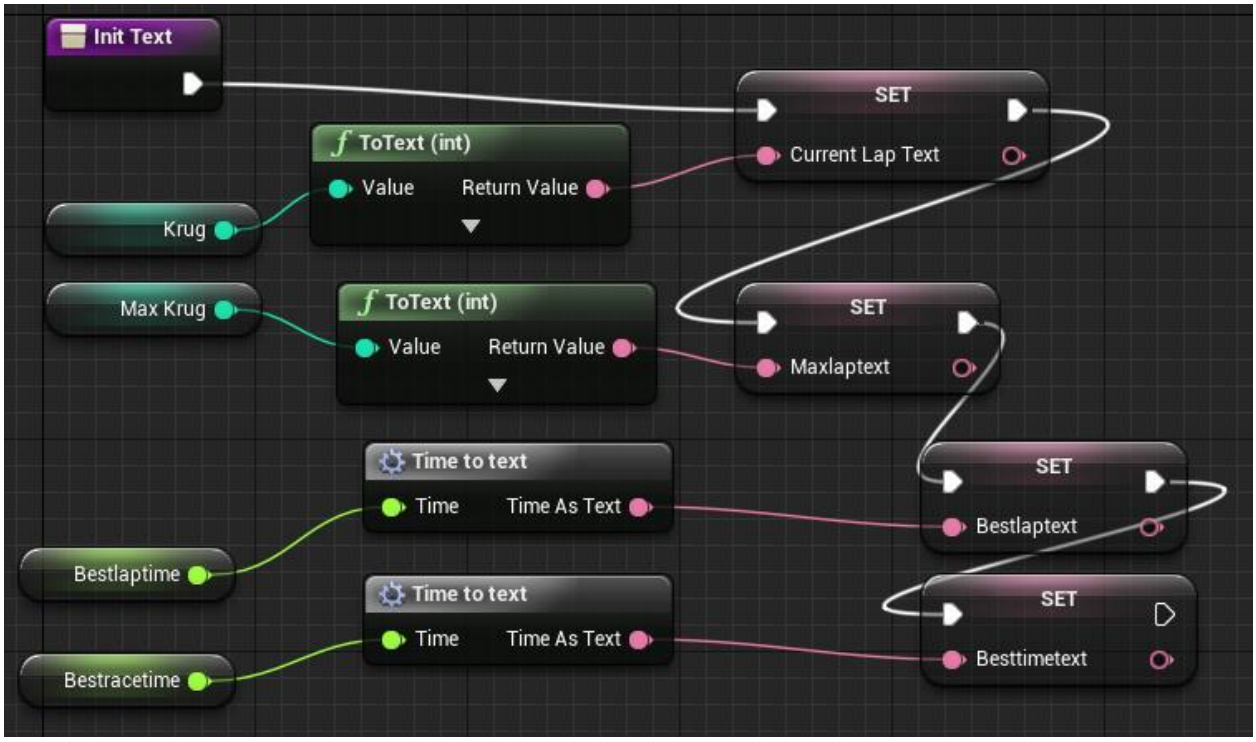


Slika 9.18 Input i output

Makro započinje sa inputom te se grana u 3 stupa. Stup za minute, sekunde i milisekunde. Svaki stup zatim uzima vrijednost vremena te ju pretvara u minute, sekunde ili minute. Stup za minute dijeli vrijednost vremena koja je u sekundama sa 60 i time se dobivaju minute. Nakon što se sve vrijednosti zaokruže, između njih se stavljaju dvotočke, te na kraju sve vrijednosti se spajaju u jednu varijablu tipa string.To će omogućiti da se sve vremenske vrijednosti koje igra ima prikažu na ekranu igraču u pravilnom formatu i obliku.

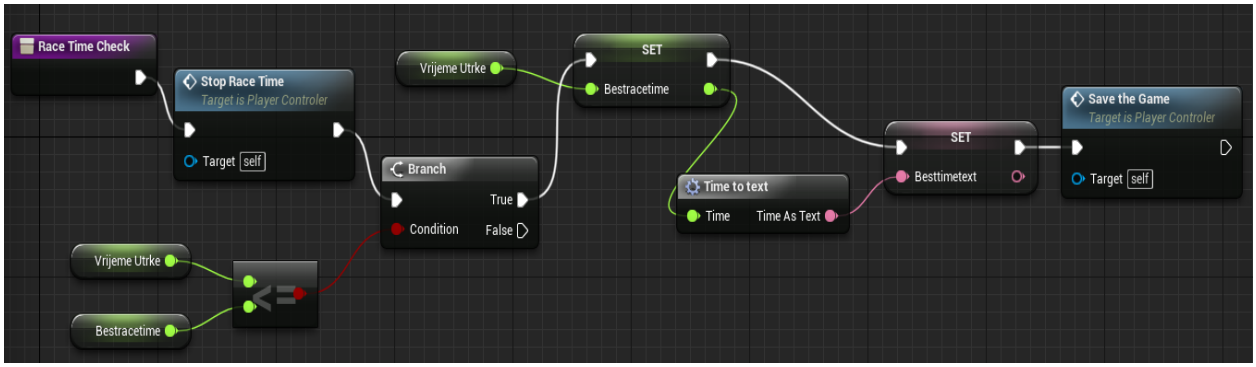


Slika 9.19 Izgled skripte



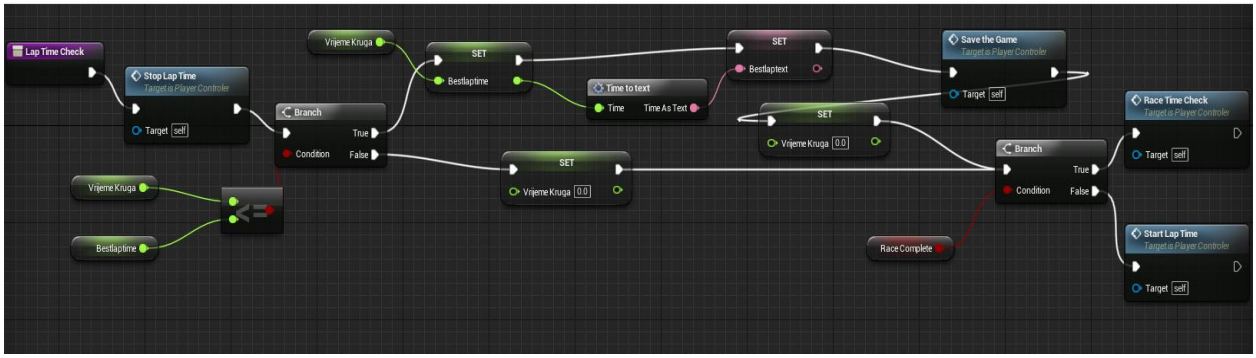
Slika 9.20 Skripta init tekst

Funkcija init Text sprema broj krugova te vremena u varijable tipa string, za vremena koja nisu cijeli brojevi koristi se novo izrađeni makro koji te iste pretvara u tekstualni oblik.



Slika 9.21 Skripta provjere vremena

Funkcija TimeCheck provjerava ako je igračevo vrijeme na kraju utrke bolje od početnog vremena. Ukoliko je manje onda igračevo vrijeme postaje novo najbolje vrijeme te se vrijeme sprema. Ovdje je također korišten blok Branch te makro naredba kako bi se vrijednost prikazala kao string.



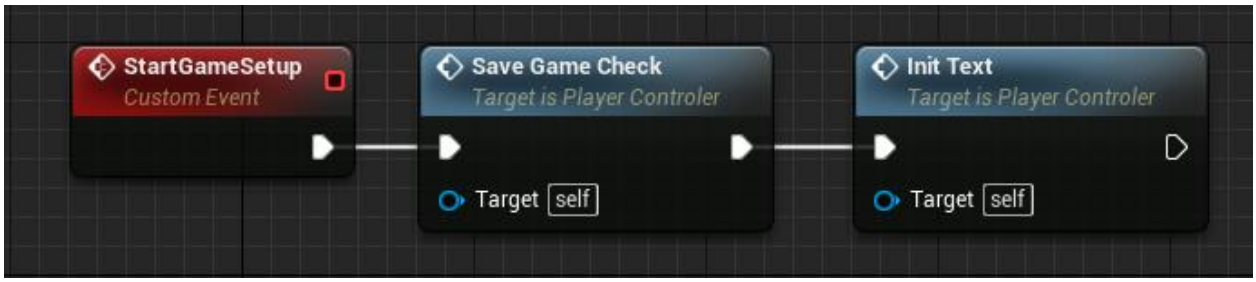
Slika 9.22 Skripta provjere krugova

Funkcija Lap Check provjerava ako je igrač postigao novo najbolje vrijeme jednog kruga te ukoliko je to istina, sprema rezultat kao novi najbolji. Kao i kod prethodne skripte provjerava se ako trenutno vrijeme kruga je manje od zadanog vremena. Ukoliko je to istina igračevo vrijeme se sprema kao novo najbolje.

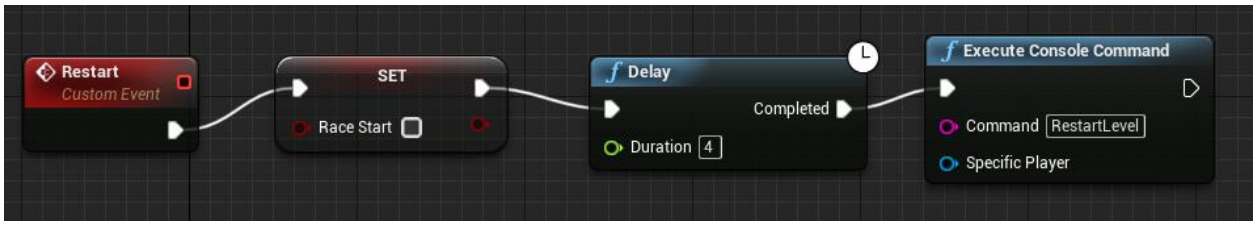
9.3 Završavanje kontrolera

Kada su sve funkcije u kontroleru obrađene potrebno ih je u event grafu pozvati i pokrenuti. Sve funkcije koje su do sada napravljene trenutno ne rade ništa dok god ih kontroler ne pozove.

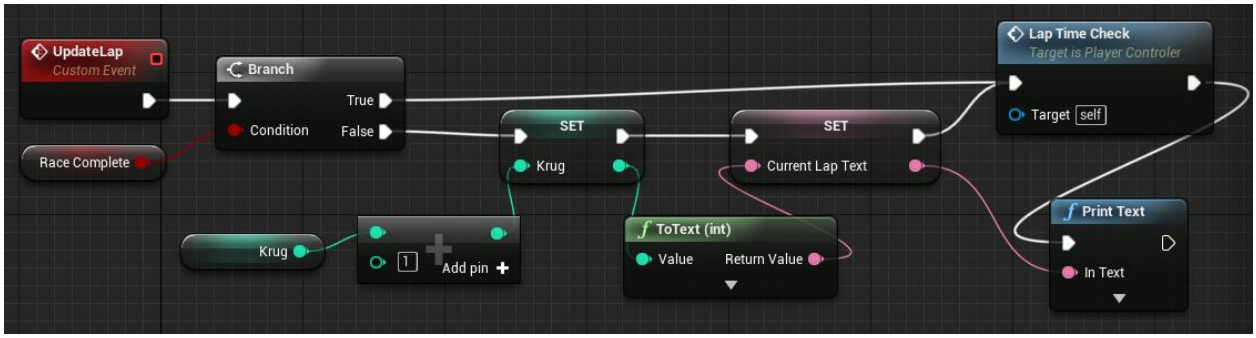
Prvi događaj koji treba pozvati je provjera ima li spremljenih igara te pokretanje funkcije koja se bavi prikazom svog teksta, zatim ponovno pokretanje igre, prikazivanje broja krugova te pokretanje timera i postavljanje početnih vrijednosti vremena ukoliko se pritisne određena tipka te skripta koja vraća igrača na prijašnju točku.



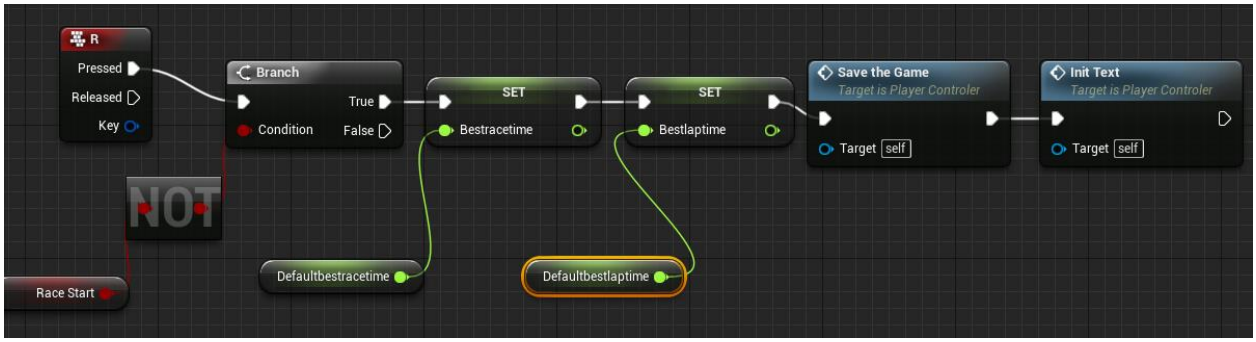
Slika 9.23 Pozivanje funkcija



Slika 9.24 Skripta restart

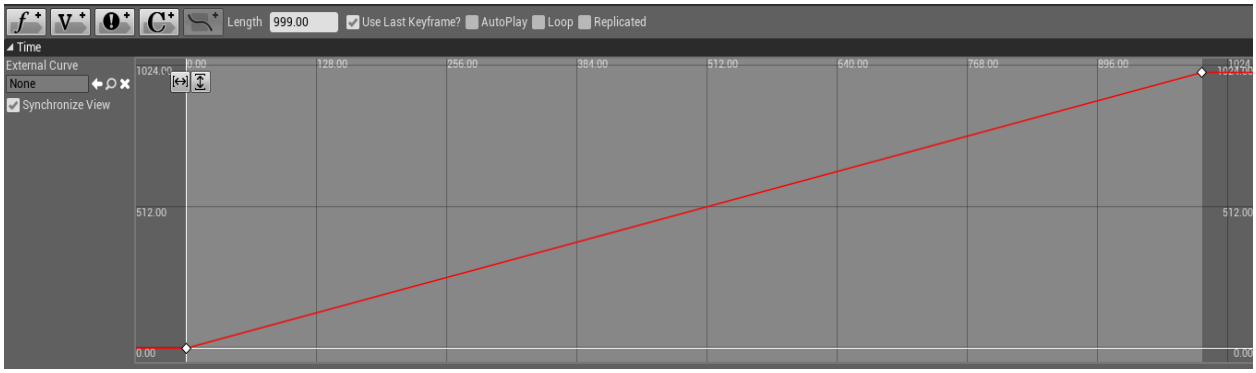


Slika 9.25 Skripta promjene krugova



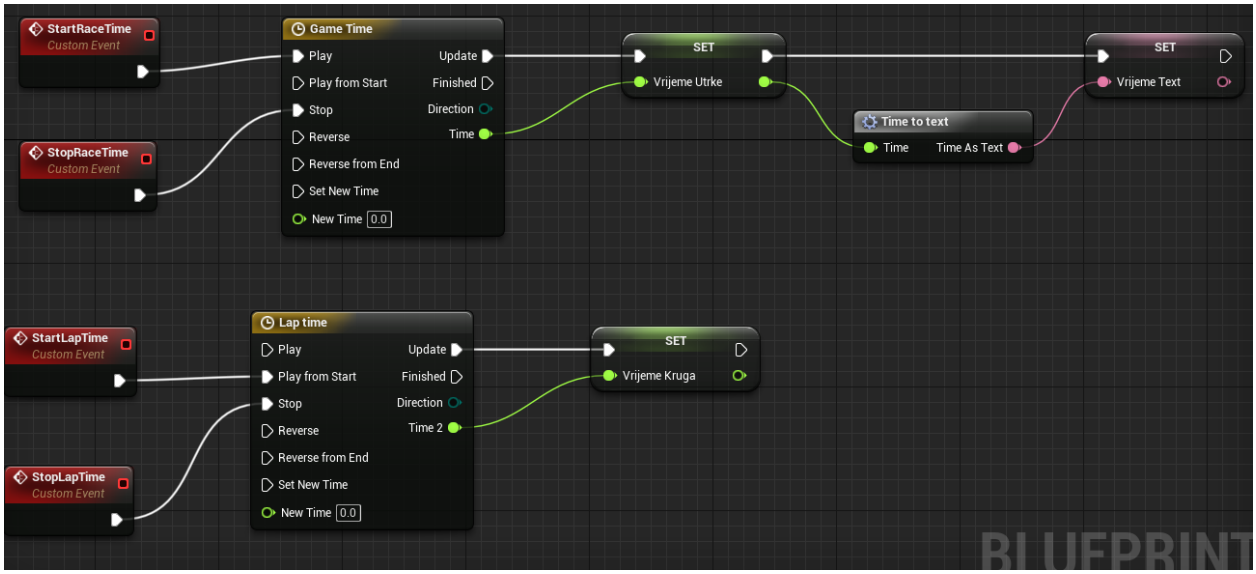
Slika 9.26 Skripta pritiska tipke R

Za pokretanje timera koji odbrojava za vrijeme utrke korišten je element Timeline. Nakon njegovog stvaranja moguće ga je otvoriti te postaviti 2 točke na vremensku liniju, jednu na početak, a drugu na neku visoku vrijednost poput 999. Važno je također označiti opciju „Use last keyframe“ iznad linije.



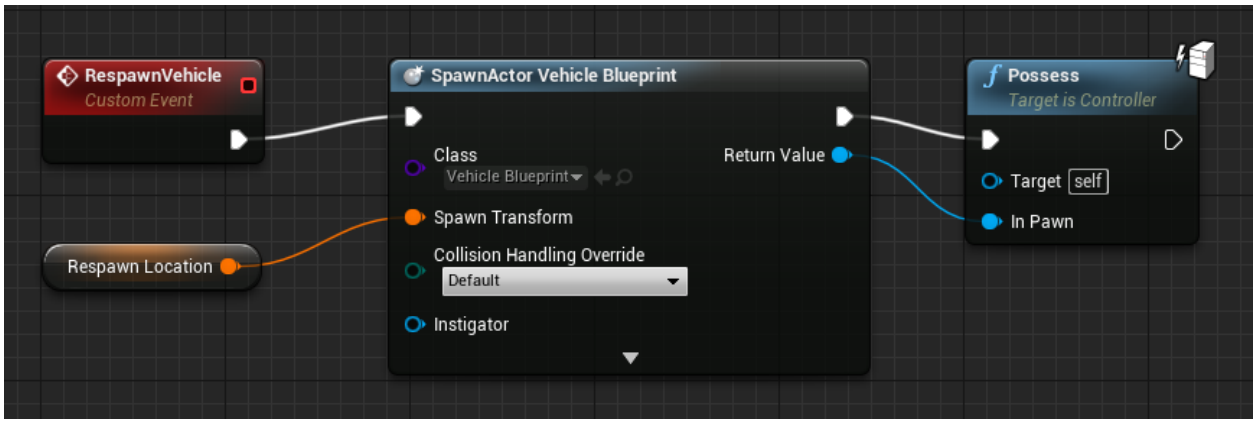
Slika 9.27 Vremenska linija

Vremensku liniju nakon izrade potrebno je duplicirati, pošto su potrebne dvije. Jedna za vrijeme kruga i jedna za ukupno vrijeme cijele utrke. Na njih je zatim potrebno spojiti događaje koji započinju i zaustavljaju odbrojavanje vremena kao i makro koji pretvara decimalni zapis u pravilni vremenski zapis na ekranu.



Slika 9.28 Skripta vremena

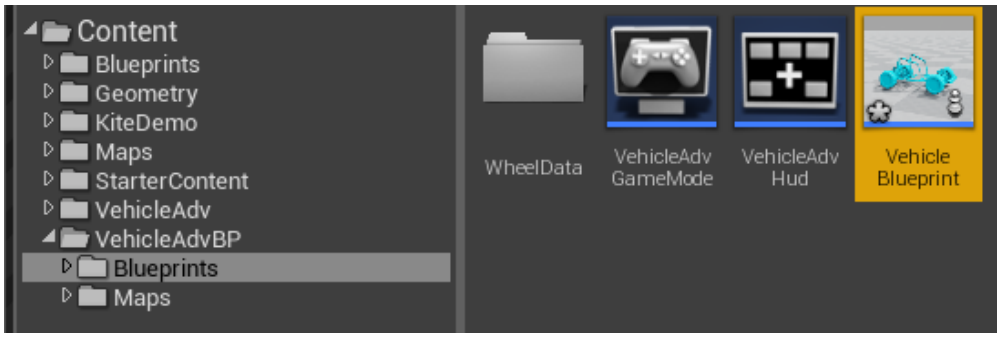
Zadnja skripta ponovno stvara igrača na mapi. Ukoliko igrač odluči da se vrati na prijašnju kontrolnu točku potrebno je uništiti njegov automobil, vratiti ga na prijašnju točku, stvoriti novi automobil te igraču dati kontrolu nad njim. Prvi dio ove skripte nalazi se ovdje u kontroleru dok drugi dio je u samom blueprintu modela automobila. Dio koji je u kontroleru stvara automobil na lokaciji kontrolne točke te igraču daje kontrolu nad njim.



Slika 9.29 Skripta ponovnog stvaranja vozila

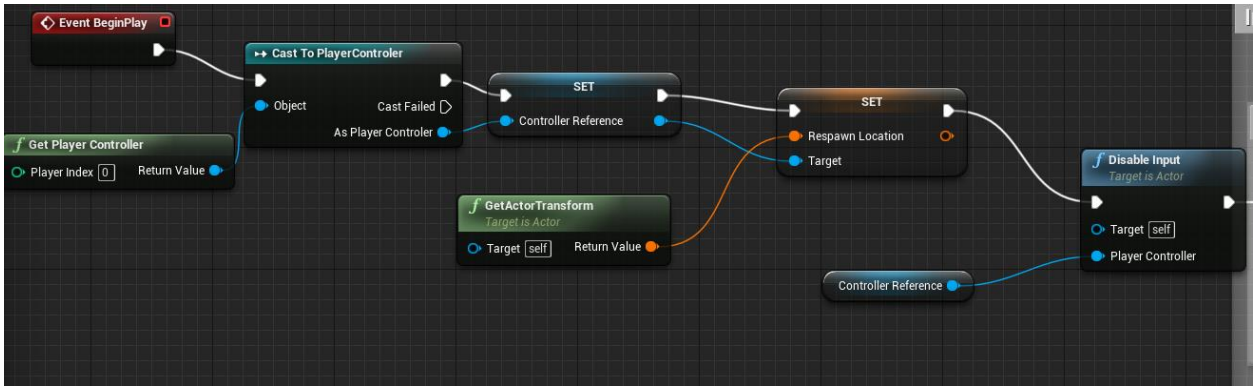
9.4 Dodavanje skripti u blueprint vozila

Za izradu skripte koja će se baviti vraćanjem igraća na prijašnju točku potrebno je otvoriti blueprint vozila iz templata koji se nalazi u datoteci VehicleAdvBP.



Slika 9.30 Lokacija blueprinta

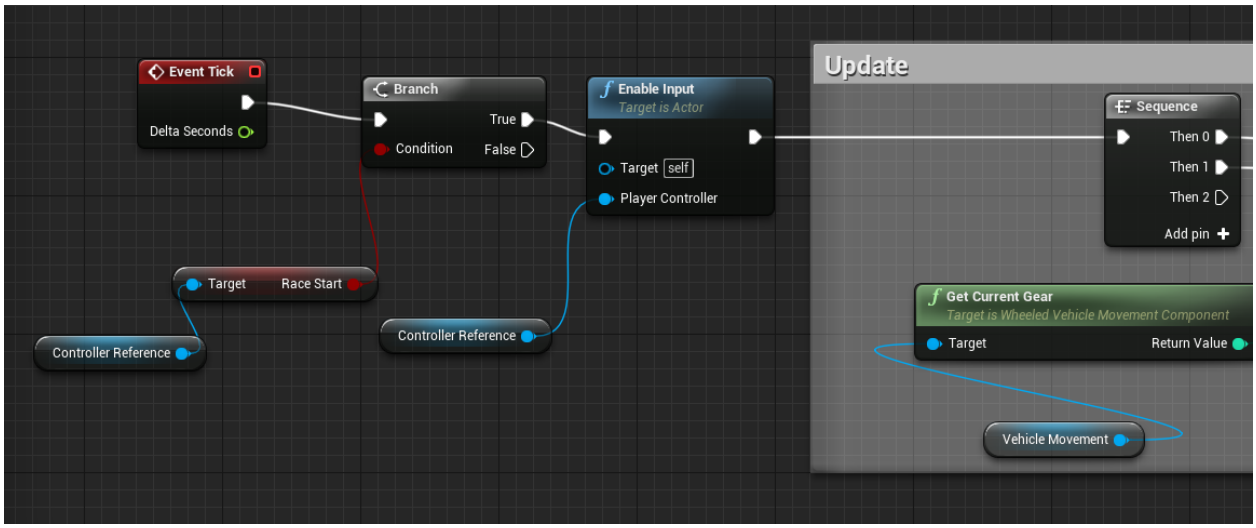
Pošto u blueprintu vozila već ima puno skripti potrebno je nove skripte ugnijezditi bez da se poremeti red kojim se one izvršavaju ili što one izvršavaju. Najbolji način za to je da se na samom vrhu doda nova skripta koja se zatim spaja na početak skripte koja je od prije tu. Početak skripte prepoznaje se po elementu Event Begin Play koji se nalazi na vrhu. Taj element se zatim odvoji od ostatka te se iza njega spoji nova skripta koja se zatim spaja u onaj element gdje je prije bio Event Begin Play.



Slika 9.31 Skripta

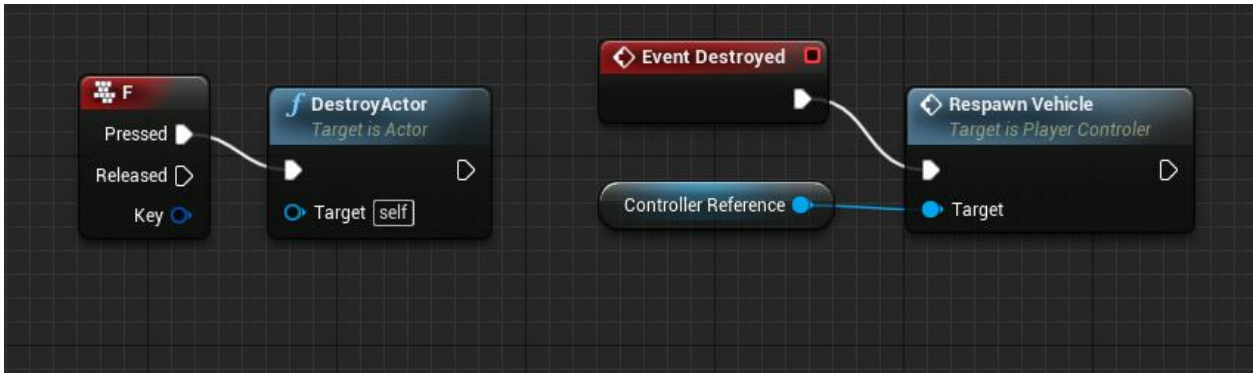
Ono što ova skripta radi je onemogućivanje kretanja igrača prije nego utrka počne, također postavlja početnu poziciju kao trenutnu kontrolnu točku u slučaju da se igrač želi vratiti prije nego uopće dođe do prve kontrolne točke. Ukoliko igrač prijeđe kroz kontrolnu točku ona će automatski postati točka na koju se može vratiti.

Kako bi se igraču vratila kontrola nad automobilom nakon što utrka počne potrebno je dodati skriptu u dio koji je nazvani Update.



Slika 9.32 Skripta vračanja kontrole

Ova skripta daje igraču ponovno kontrolu nad vozilom nakon što odbrojavanje završi i utrka počne. Korišten je blok Enable input.



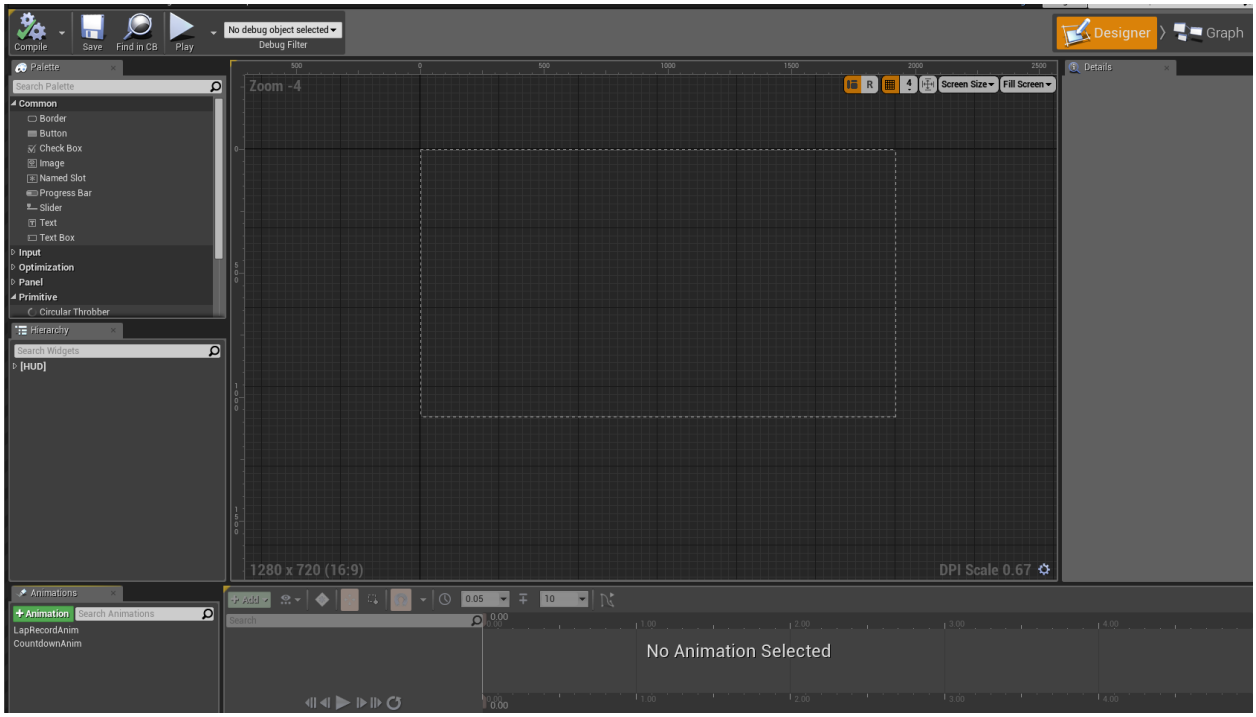
Slika 9.33 Skripta uništavanja igrača

Na kraju je dodana skripta koja uništava igrača ukoliko je pritisnuta tipka F. Kraj toga nalazi se još jedna mala skripta koja ponovno stvara igrača ukoliko je uništen. Sa njih dvije zajedno igrač pritiskom na tipku F vraća se na prijašnju kontrolnu točku.

10. Izrada grafičkog sučelja

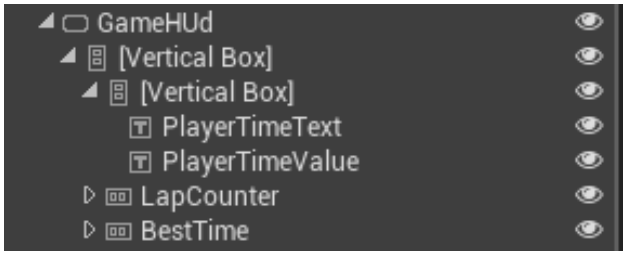
Za izradu grafičkih sučelja Unreal engine ima integrirani editor pod imenom Motion Graphics.[16] On ima sve opcije koje su nužne da se izradi korisničko sučelje igre, početni ekrani, glavni izbornici i slično. Za svrhe ove igre na glavnom ekranu trebaju pisati vremena koja igrač treba pobijediti, također i upute kako igru pokrenuti. Nakon što se igra pokrene početni izbornik treba se maknuti, a umjesto njega treba doći sučelje koje će biti vidljivo dok igrač vozi. Na njemu treba biti vrijeme kako bi igrač znao koliko vremena već vozi, kao i vrijeme od kojeg mora biti brži zajedno sa brojem krugova koje još ima.

Sama izrada sučelja dijeli se u 2 dijela, prvi dio je dizajnerski, gdje se na platnu stavljaju slike, tekst i svi ostali elementi koji su vidljivi, te programski dio koji funkcionira kroz blueprinte. Što se tiče samih elemenata oni su ponuđeni u gornjem lijevom kutu te se ih dodaje povlačenjem u hijerarhiju.

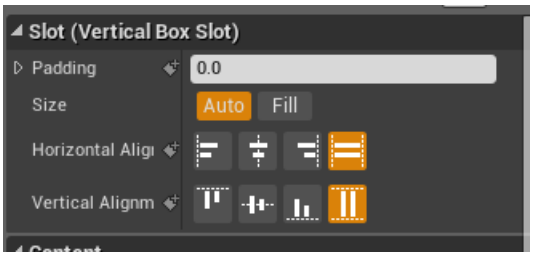


Slika 10.1 Prozor za izradu sučelja

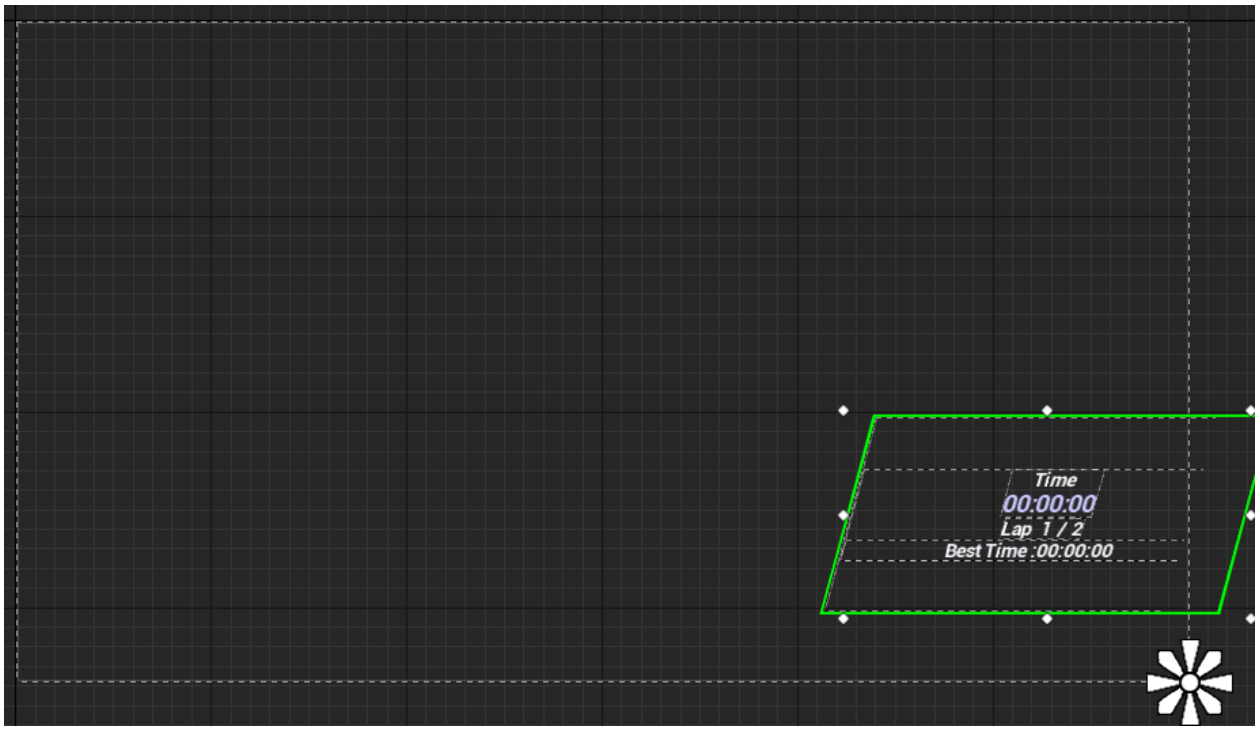
Za izradu grafičkog sučelja prvo je potrebno dodati Border, a nakon toga Vertical Box u hijerarhiju kao dijete Borderu. Kako bi se dodao tekst potrebno je povući element tekst ispod u sam Vertical Box. Za centriranje i poravnanje koriste se opcije vertikalnog i horizontalnog poravnanja u desnom dijelu, također tamo se nalaze i opcije mijenjanja boje fonta, vrste fonta te veličine fonta.



Slika 10.2 Elementi



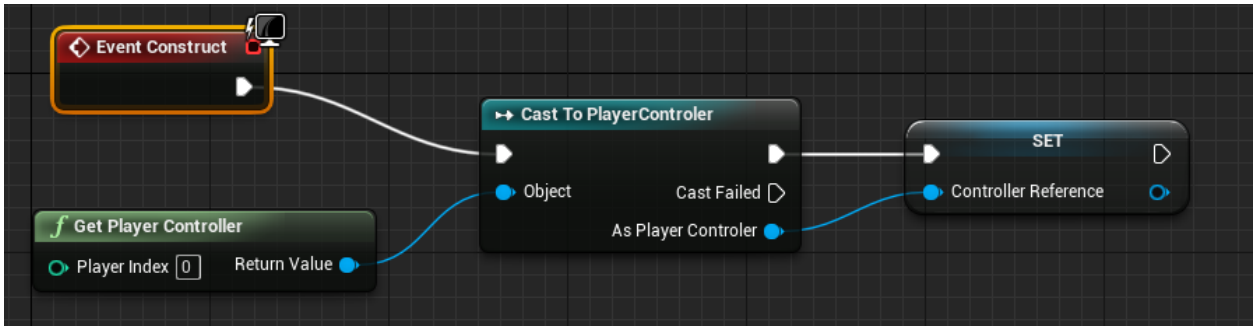
Slika 10.3 Opcije poravnanja



Slika 10.4 Platno

Nakon izrade svih elemenata i njihovog centriranja cijeli blok se može pomicati i staviti u bio koji dio platna.

Prije izrade glavnog izbornika potrebno je stvoriti referencu kontrolera unutar programskog dijela sučelja kako bi vrijednosti koje se vide na ekranu odgovarale onim stvarnim vrijednostima. Za to je potrebno otvoriti Graph u gornjem desnom kutu te sastaviti malu skriptu koja to omogućava.

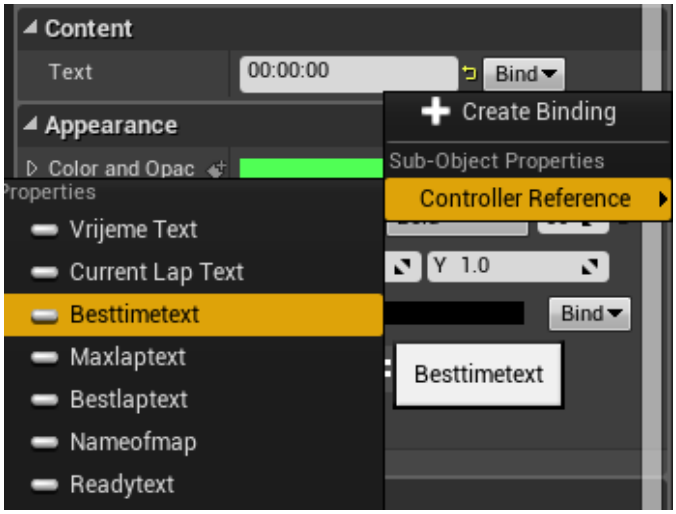


Slika 10.5 Skripta za sučelje

Nakon toga potrebno je stvoriti i nekoliko varijabli koje su potrebne za sakrivanje elemenata sučelja. Za razliku od ostalih varijabli, ovdje je tip varijable ESlate Visibility. Ove varijable omogućavaju da se vidljivost sučelja kontrolira preko skripti, najbolji primjer toga je glavni izbornik koji mora nestati kada igra započne. Njegovo nestajanje se odrađuje tako da se varijabli koja je njemu zadana promijeni vrijednost iz „Visible“ u „Hidden“. U ovom slučaju jedino SplashScreenVisibility ima vrijednost „Visible“, ostale dvije varijable imaju vrijednost „Hidden“.

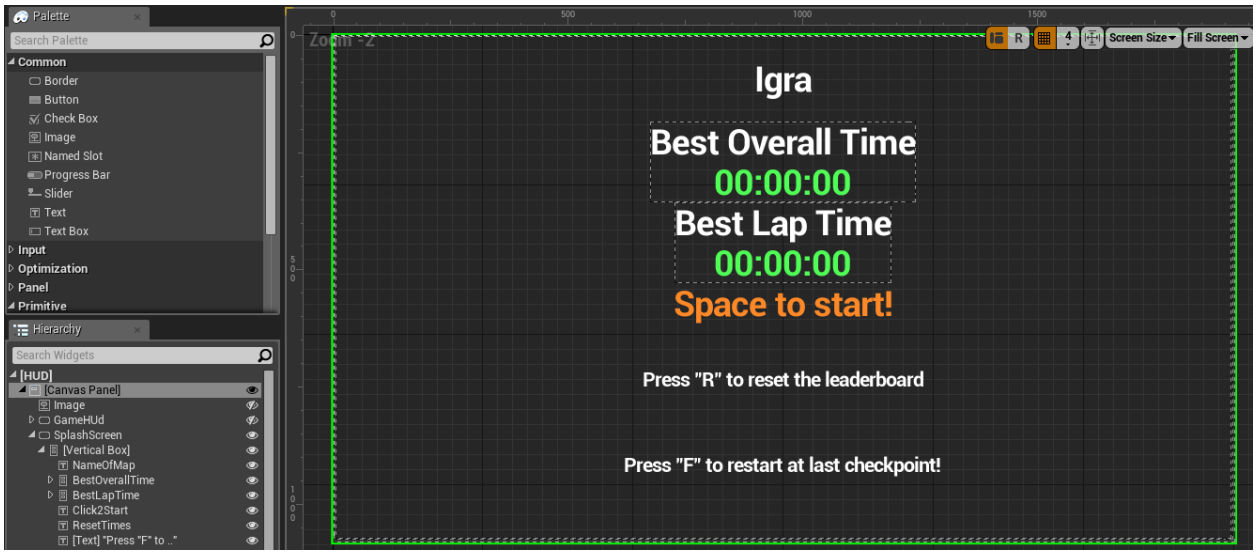
Slika 10.6 Varijable sučelja

Na dizajnerskom dijelu zbog prethodne skripte pojavila se nova opcija kraj vrijednosti vremena koja je trenutno ručno upisana. Opcija se zove Bind, a ono što ona omogućuje je spajanje vrijednosti napravljenih varijabla i elemenata grafičkog sučelja.



Slika 10.7 Opcija Bind

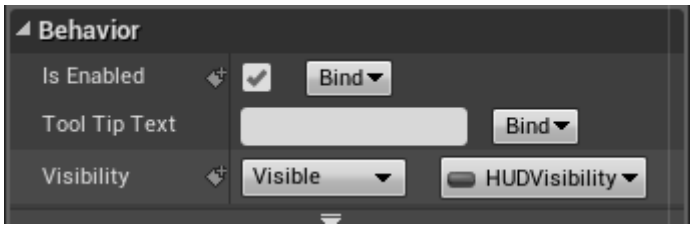
Ovisno na koju varijablu se tekst spoji, njenu vrijednost će prikazivati. Ovaj postupak se odradi za sve vrijednosti na sučelju za koje postoje varijable, u ovom slučaju to su sva vremena, ime igre, ime mape, te tekst koji će služiti za odbrojavanje prije utrke.



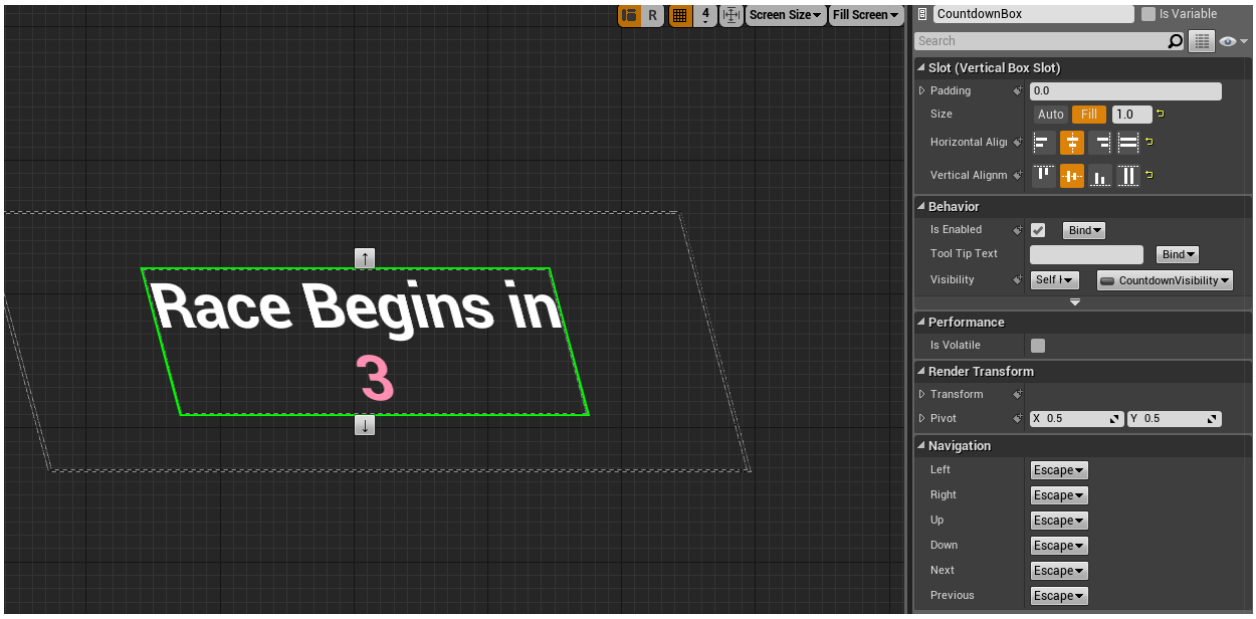
Slika 10.8 Izgled sučelja

Nakon izrade sučelja unutar igre izrađuje se glavi izbornik koristeći iste alate kao i kod prijašnjeg sučelja. Također sve vrijednosti spajaju se na pripadajuće varijable.

Osim spajanja varijabli sa vrijednostima potrebno je spojiti i varijable koje nose informacije o vidljivosti elementa.[18] To se radi pritiskom na gumb Bind kraj opcije Visibility te odabirom pripadajuće varijable vidljivosti.



Slika 10.9 Opcija Visiblity



Slika 10.10 Izgled odbrojavanja

Zadnji dio sučelja je dio sa odbrojavanjem prije same utrke, u ovom slučaju broj je potrebno povezati za varijablom, a cijeli element i sa pripadajućom varijablom vidljivosti. Ovaj element će biti drugi u redu koji će se pojaviti na ekranu, prvi je glavni izbornik, a treći je sučelje koje prati igrača.

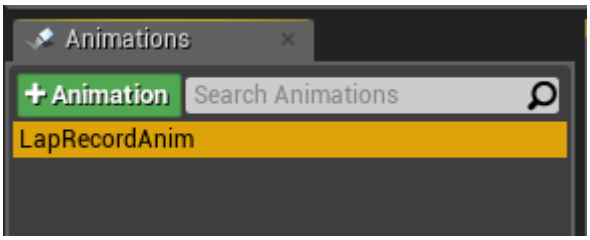
10.1 Animacija

Unutar samog editora moguće je izrađivati animacije sa sučeljem, to znači da sučelje ne mora biti statično već može biti bogato sa animacijama.[17] Animacije je najbolje držati u granicama normale inače sučelje postane previše zatrpano sa stvarima koje korisniku odvlače pažnju i smetaju. U slučaju ove igre animacija je korištena kako bi se pojavilo upozorenje da je igrač pobijedio najbolje vrijeme kruga.

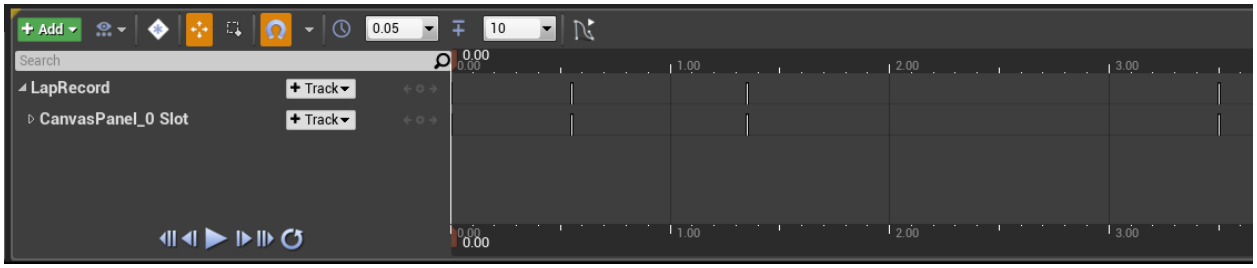
Pošto je upozorenjeposebni element za izradu je dovoljno kopirati jedan od prethodno izrađenih elemenata.

Slika 10.11 izgled teksta

Zatim je potrebno izraditi novu animaciju pritiskom za tipku Animation. Novu animaciju je potrebno nazvati, a selektirajući animaciju pojavljuje se vremenska crta na dnu ekrana.



Slika 10.12 Animacija



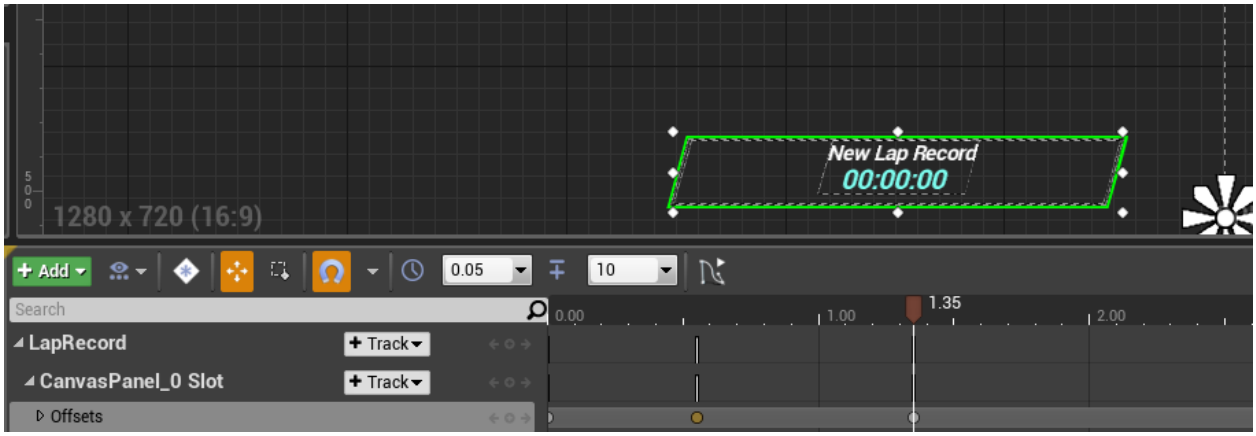
Slika 10.13 Vremenska crta

Za animiranje teksta potrebno ga je prvo selektirati, zatim spojiti vrijednosti teksta sa varijablom istog imena. Nakon toga tekst treba povućiizvan granica platna, te pritisnuti na malu ikonu koja se nalazi s lijeve strane vrijednosti. Ta ikona radi točku na vremenskoj crti (eng. Keyframe) koja pamti poziciju elementa kroz vrijeme.

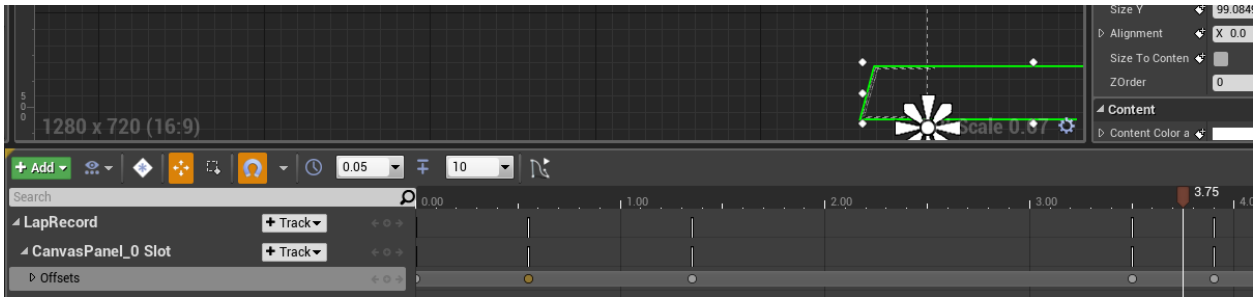
Slika 10.14 Vrijednost pozicije na X osi

Nakon prve točke potrebno je pomaknuti vremensku liniju nekoliko sekundi unaprijed, ovisno o tome koliko dugo element treba putovati te ga pozicionirati na novu poziciju i pritiskom na istu ikonu spremiti njegov položaj na vremenskoj crti.

Isti postupak se ponavlja i za zadnju točku na kojoj se element ponovno vraća izvan platna.

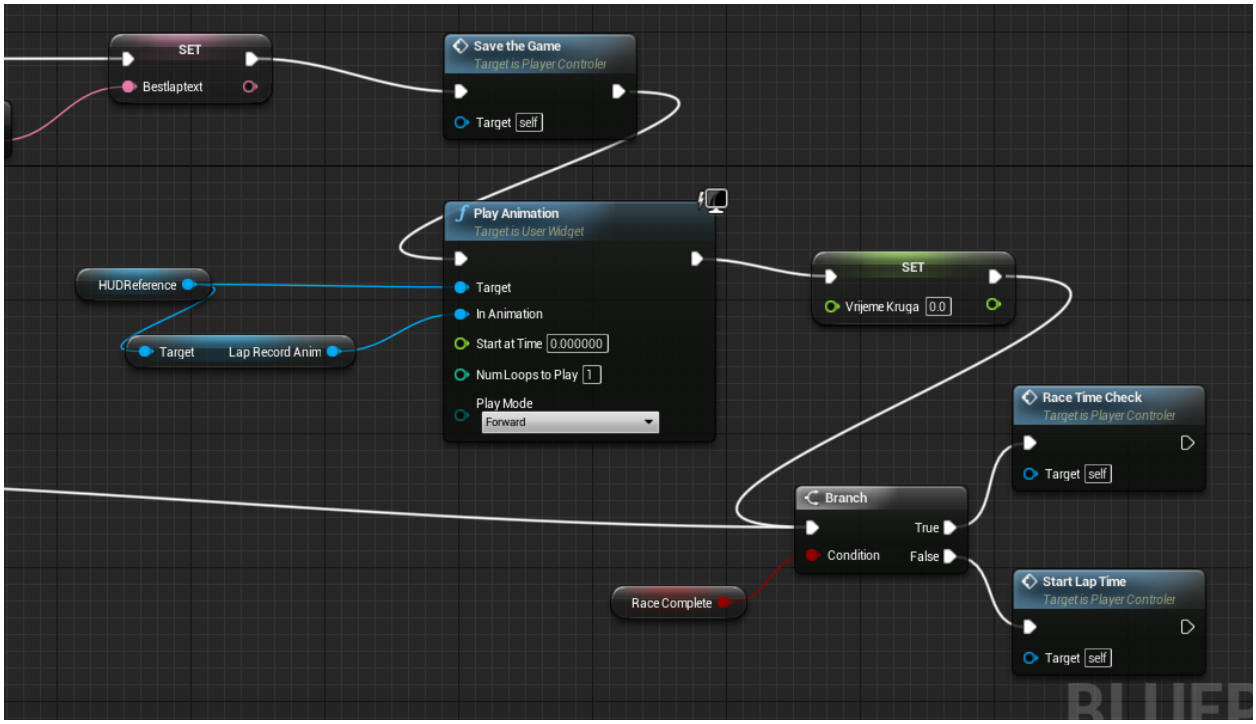


Slika 10.15 Animacija dolaska teksta



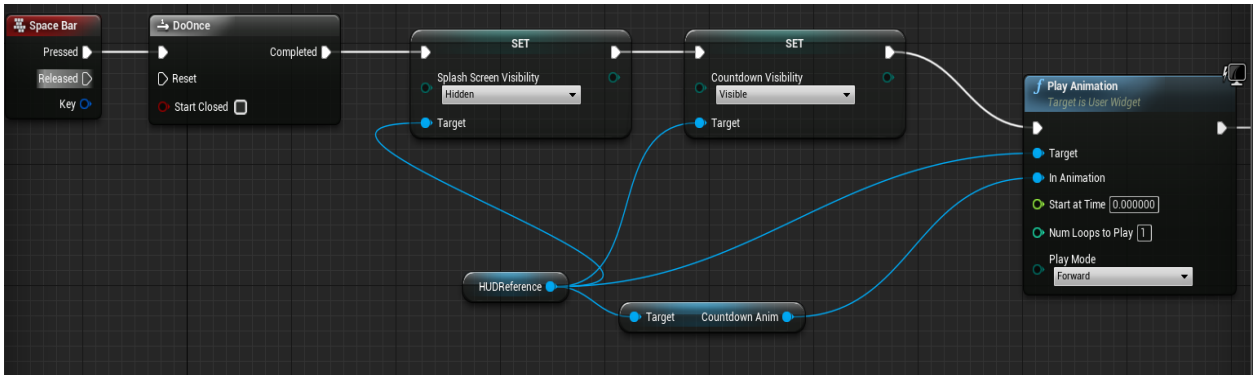
Slika 10.16 Animacija odlaska teksta

Kako bi animacije radile za vrijeme igre zajedno sa cijelim korisničkim sučeljem potrebno je dodati ih kroz skriptu. Za animaciju novog rekorda kruga potrebno je otvoriti funkciju koja provjerava vrijeme kruga unutar kontrolera te dodati Play Animation element pred kraj skripte.



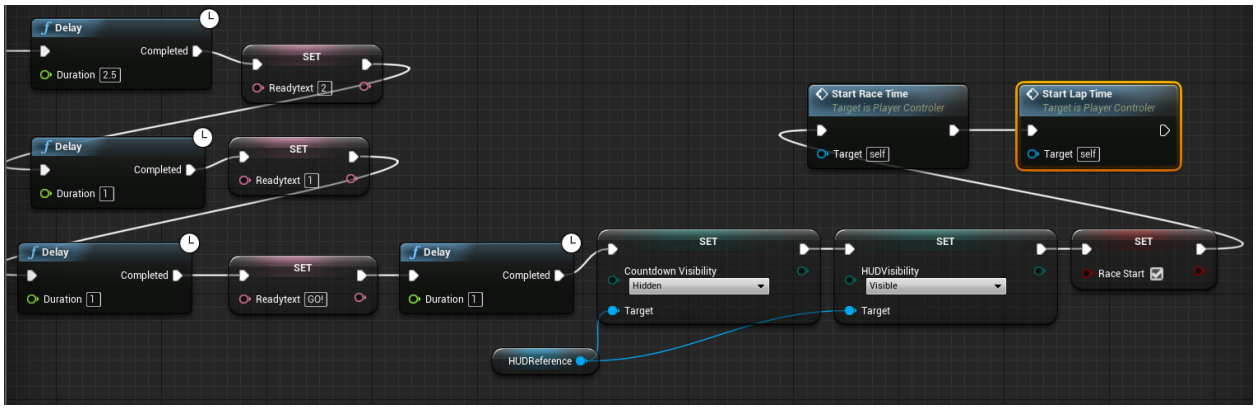
Slika 10.17 Dodavanje animacija u skriptu

Osim animacija potrebno je još spojiti glavni izbornik sa skriptama kako bi pravilno funkcionirao. Trenutno nedostaje skripta koja pokreće igru pritiskom na tipku, ta skripta nalazit će se u grafu kontrolera.



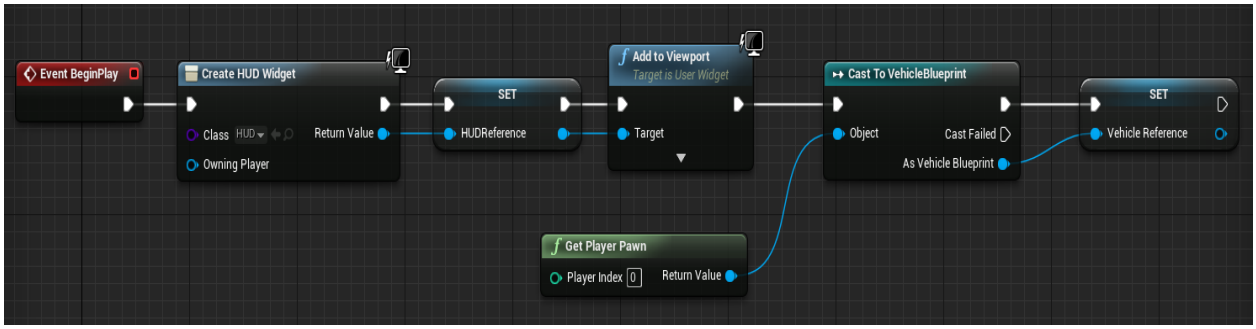
Slika 10.18 Dodavanje izbornika sa skriptom

U ovoj skripti važno je da se nakon pritiska tipke zove funkcija DoOnce koja izvršava ostatak skripte samo jedan put, inače bi svaki put kada korisnik pritisne tipku utrka ponovno počela bez da prije završi. U najgorem slučaju cijela igra bi se srušila.



Slika 10.19 Drugi dio skripte

Drugi dio skripte radi na odbrojavanju prije same utrke te sakriva elementa odbrojavanja po njegovom završetku, a pokazuje elemente sučelja namijenjenim igraču.



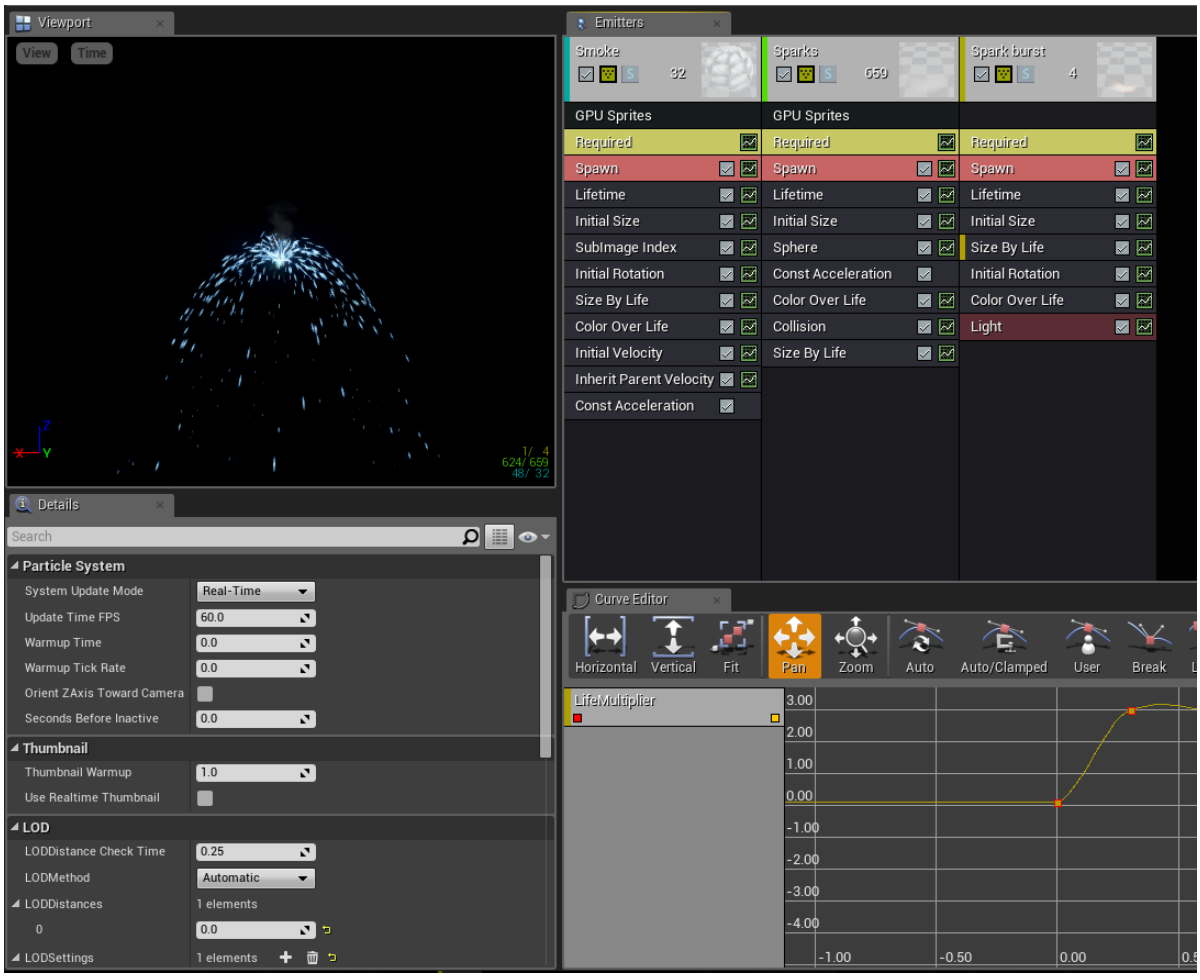
Slika 10.20 Skripta pokretanja sučelja

Završna skripta nalazi se na početku kod elementa BeginPlay. Na taj element spajaju se funkcije koje stvaraju i pokreću sučelje kada igra započne. Ukoliko ove skripte nema igra i dalje radi no sučelje se uopće ne pojavljuje zato je važno da se sučelje inicijalizira prije nego što ostatak igre počne.

11. Sustav čestica

Zadnji korak u izradi kontrolnih točaka je dodavanje sustava čestica. Pošto je element sustava dodan prije dodavanje čestica je vrlo lagano. Sve što je potrebno je učitati sustav čestice a ostalo je sve odrađeno samo. Same čestice se mogu izraditi ili urediti unutar samog engina. Za ovu igru izrađen je plavi krug čestica, čestice su uzete iz paketa koji je dodan kod stvaranja projekta. Prozor koji se otvori kod uređivanja čestica naziva se Cascade.[19]

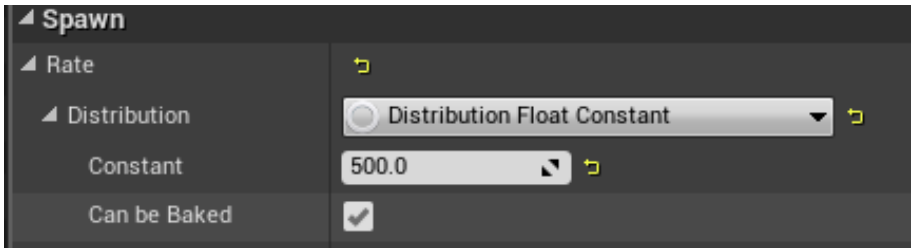
Za izradu čestica najbolje je duplicirati jedan od postojećih efekata te ga modificirati po želji.Za ovaj sustav izabran je sustav čestica pod nazivom sparks koji se nakog dupliciranja može otvoriti.



Slika 11.1 Prozor uređivanja čestica

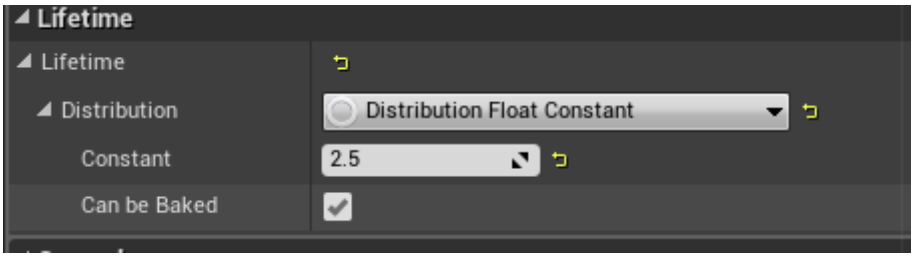
U gornjem desnom dijelu ekrana nalaze se dijelovi ovog sustava, u ovom slučaju „Smoke“ nije potreban te ga se može izbrisati klikom i pritiskom na Delete na tipkovnici.

Nakon toga prva opcija koju je potrebno promijeniti nalazi se unutar elementa Sparks pod modulom Spawn. Opcija se zove Distribution. Ona govori koliko čestica će se stvarati u efektu. U ovom slučaju ta vrijednost je 500.



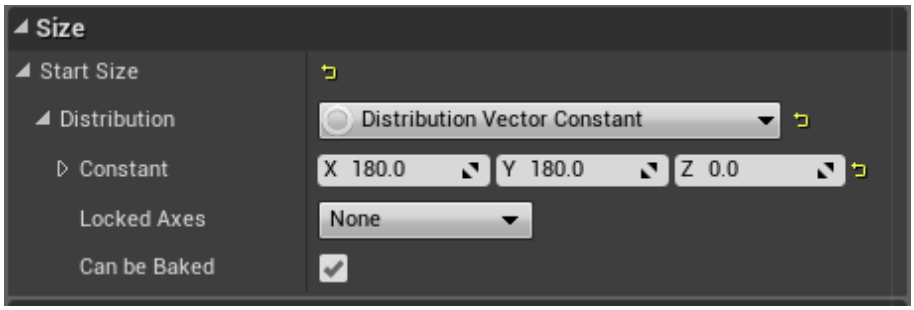
Slika 11.2 Opcija distribucije

Druga opcija nalazi se u modulu Lifetime, a regulira opcije vezane uz ponašanje efekta kako vrijeme prolazi.



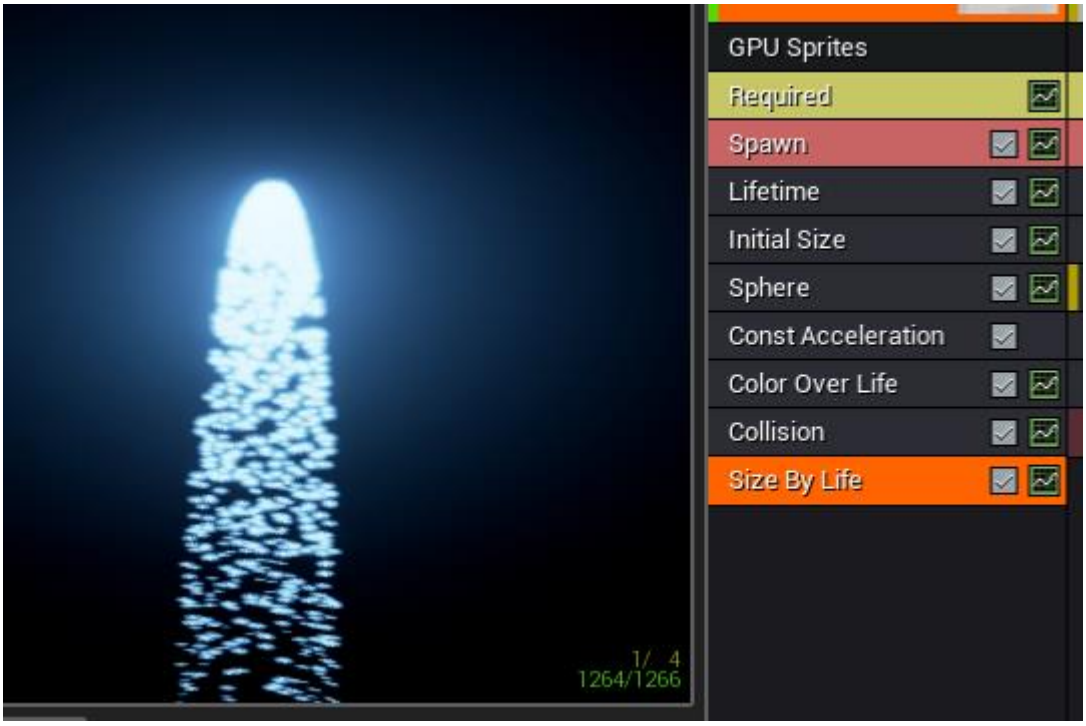
Slika 11.3 Opcija Lifetime

U modulu Initial size potrebno je promijeniti veličinu za željeni efekt kao ii distribuciju u Vektor konstantu.



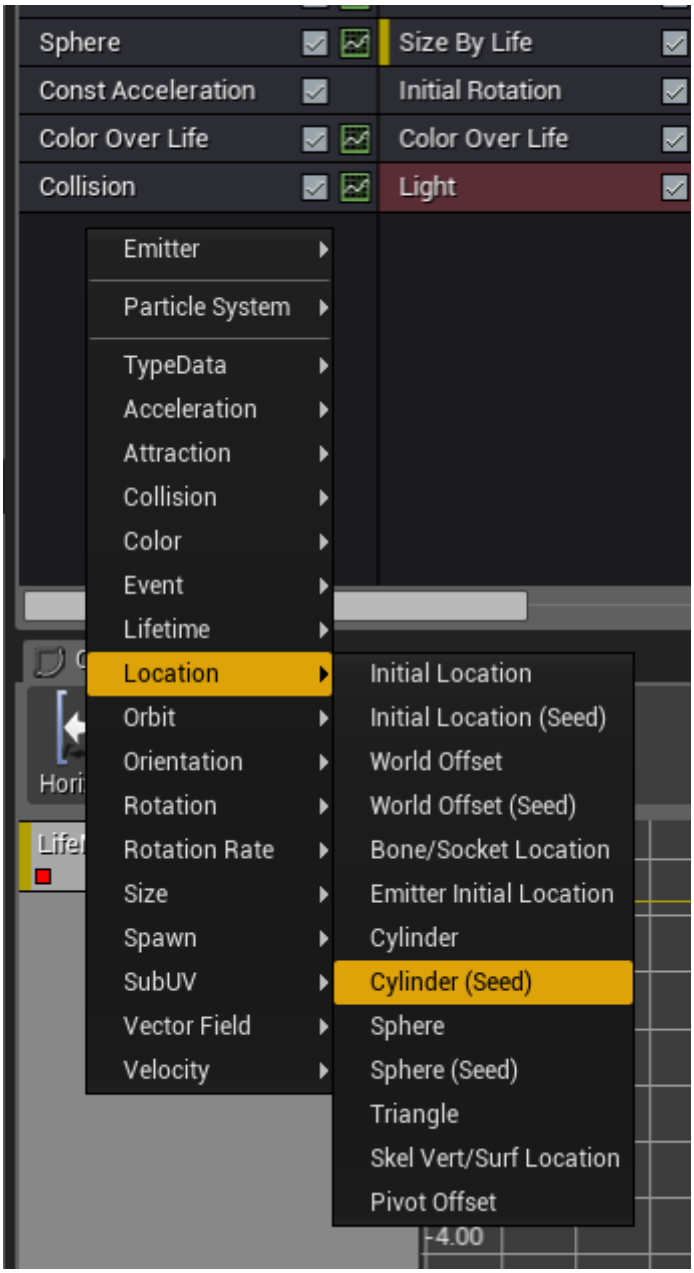
Slika 11.4 Opcija veličine

U listi modula je potrebno izbrisati modul Size by life koji baš kao što mu i ime kaže mijenja veličinu efekta kako vrijeme prolazi, taj efekt nije potreban te ga je potrebno izbrisati.



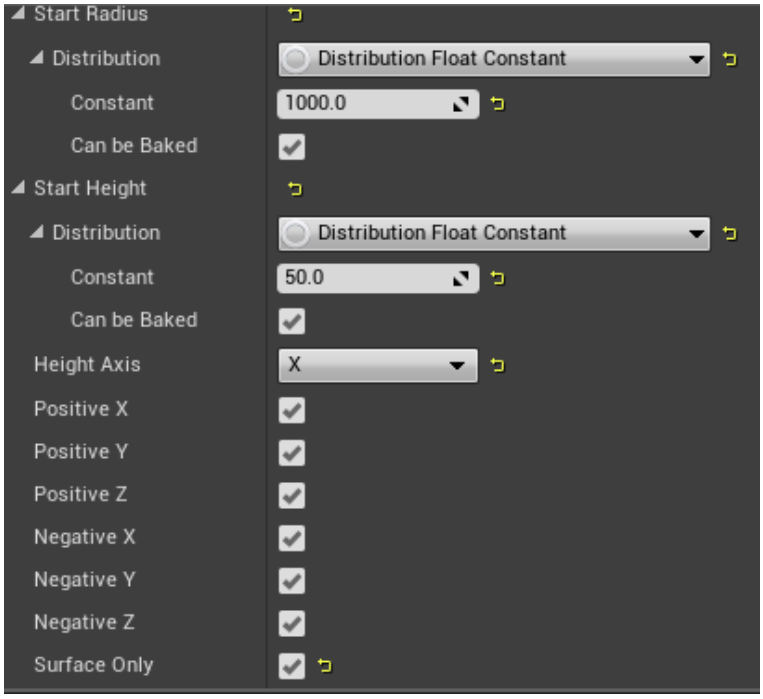
Slika 11.5 Izgled čestica

Zadnji korak je dodavanje novog modula, modul se dodaje desnim klikom u prazni prostor te pritiskom desne tipke miša, modul potreban zove se Cylinder(seed).

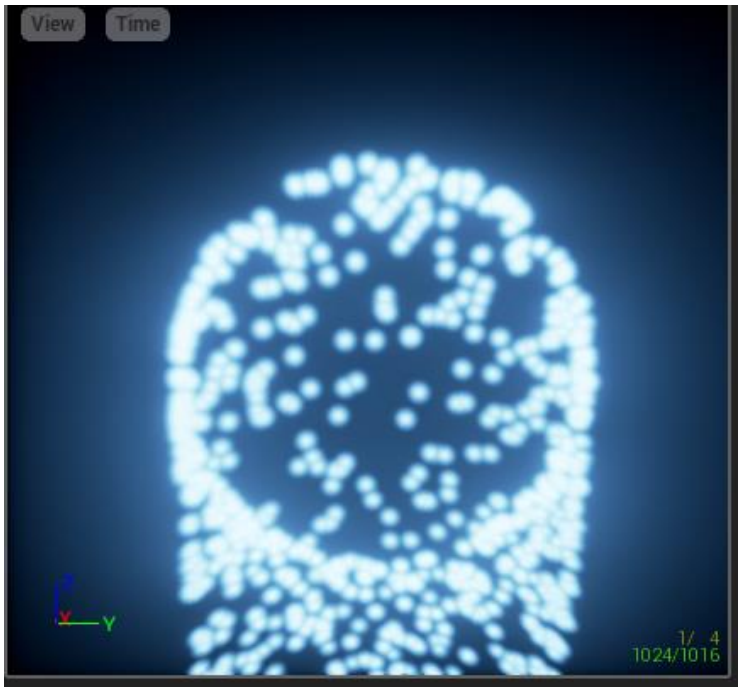


Slika 11.6 Novi modul u izborniku

U modulu opcije distribucije su promijenjene na konstante, a označena je i opcija Surface Only koja prikazuje čestice samo na rubovima tj. sredina efekta je prazna dajući efekt plamenog kruga.



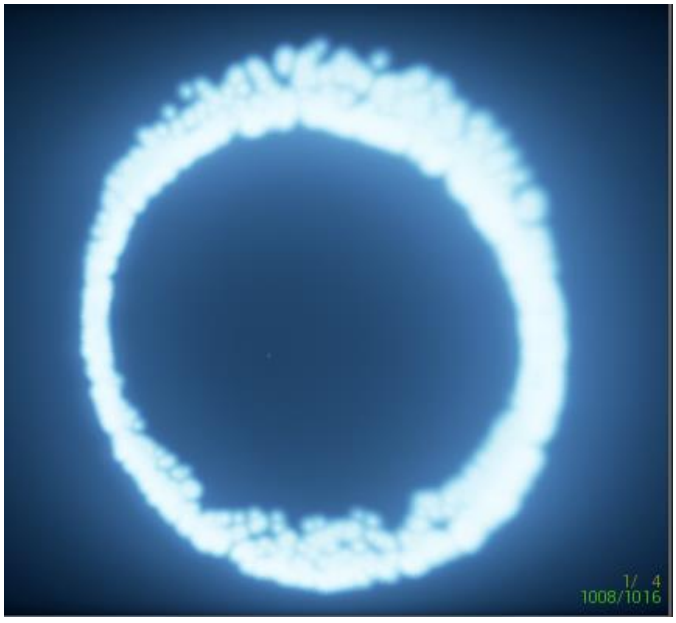
Slika 11.7 Opcije novog modula



Slika 11.8 Izgled čestica

Kako bi se maknuo efekt padanja čestica prema dolje potrebno je promijeniti vrijednost njihove akceleracije na Z osi. U ovom slučaju nova vrijednost je 150, što znači da čestice blago idu prema gore suprotno smjeru gravitacije.

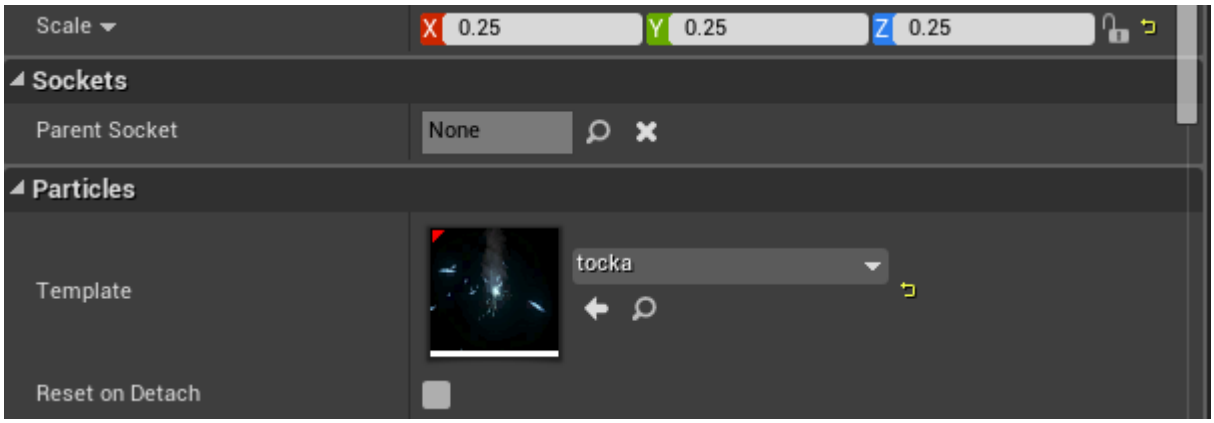
Slika 11.9 Opcija gravitacije čestica



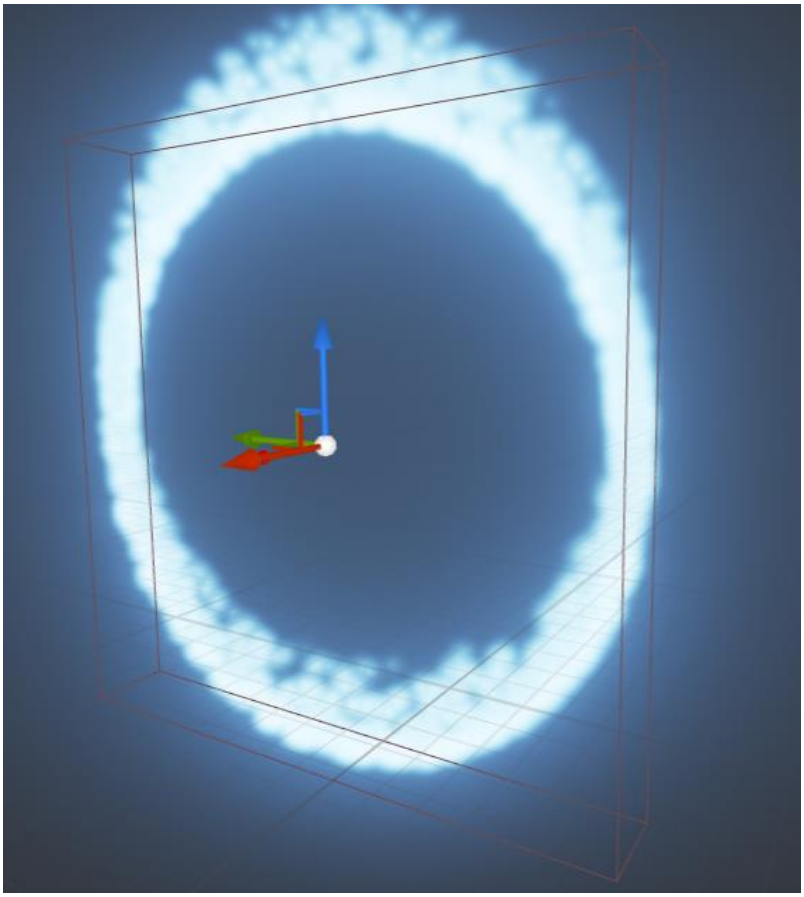
Slika 11.10 Finalni izgled čestica

Za aktivaciju čestica na kontrolnim točkama potrebno je otvoriti blueprint kontrolnih točaka te pod Template staviti novo izrađen sustav čestica. Sustav je

također potrebno i smanjiti na vrijednost kao što je 0.25 pošto mu je standardna veličina prevelika.

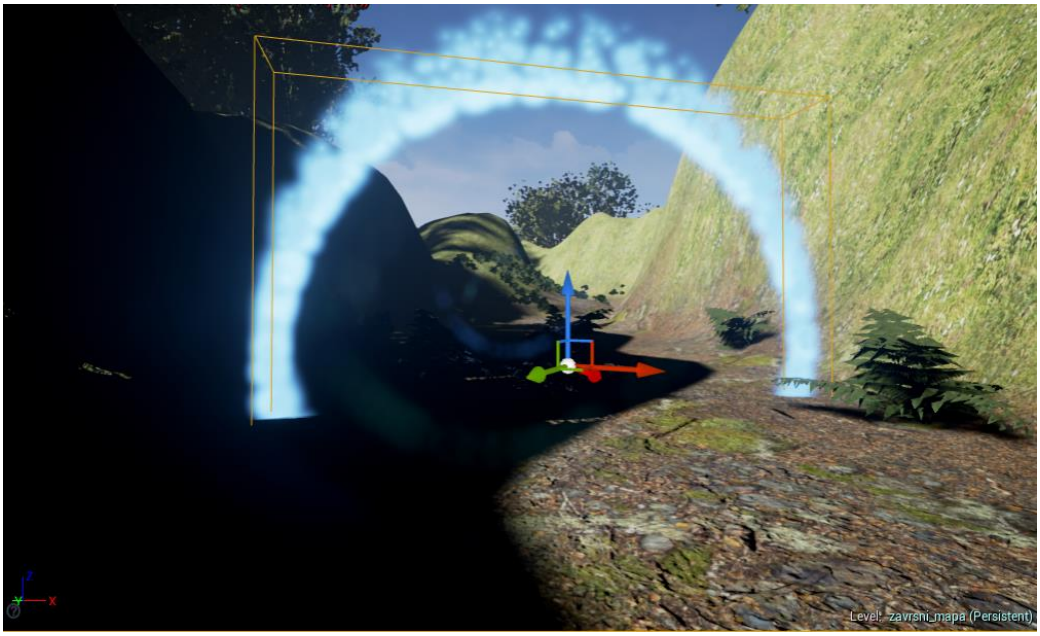


Slika 11.11 Učitavanje čestica



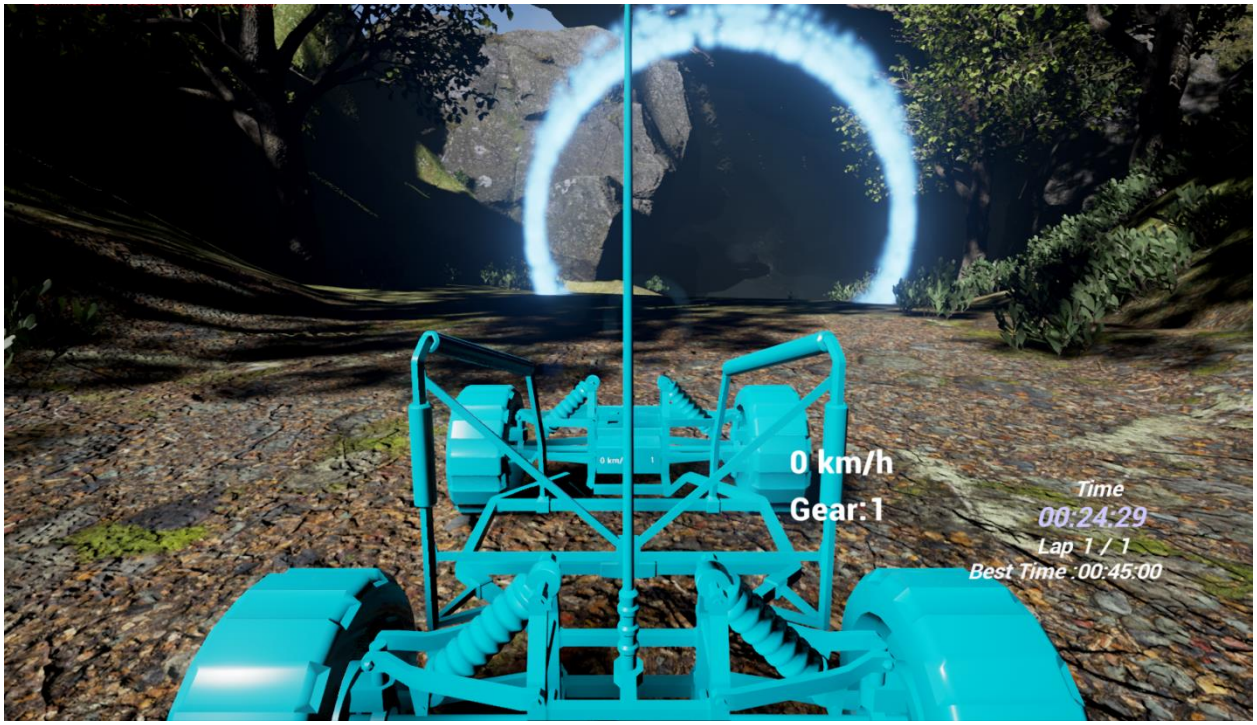
Slika 11.12 Izgled čestica na kontrolnoj točci

Kontrolne točke sada su završene i potrebno ih je postaviti u mapu na željene lokacije te dodati u listu selektirajući element sa opcijama. To će spremiti sve kontrolne točke u listu, a skripte će odraditi sav posao. Bitno je također da sve kontrolne točke nemaju označenu opciju Hidden ingame, pošto skripta ih sakriva automatski potrebno je da su one sve na početku vidljive.

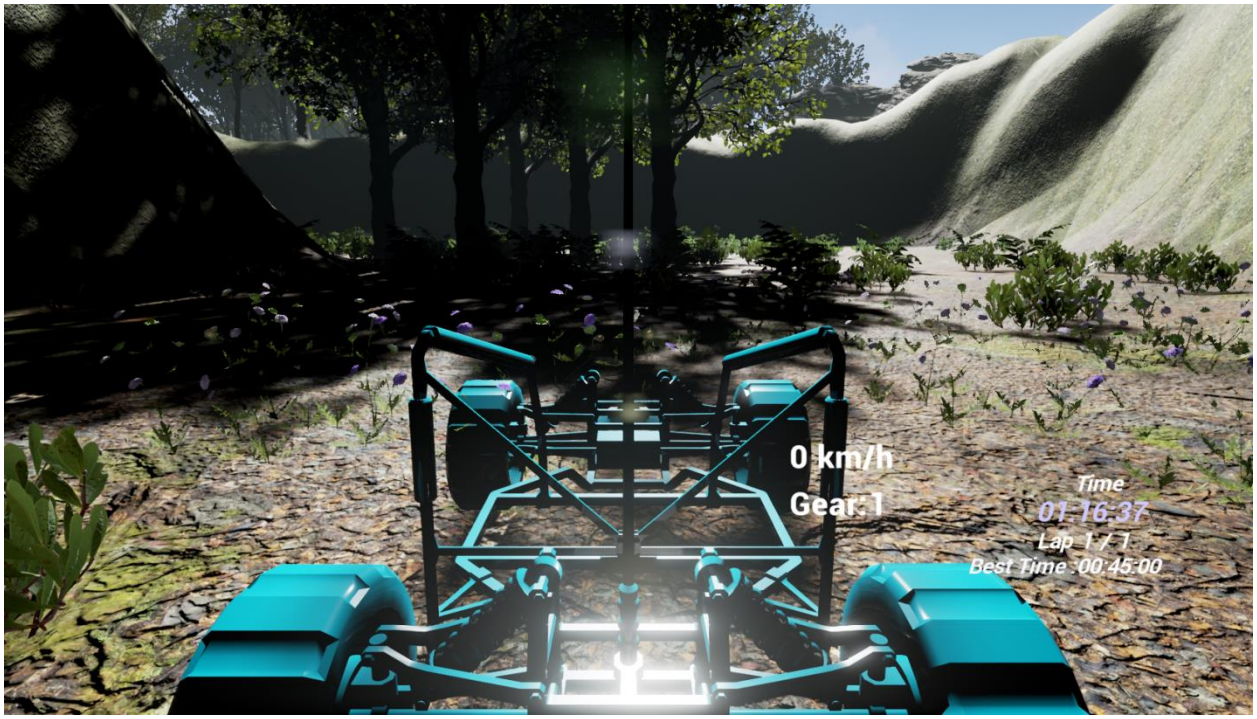


Slika 11.13 Izgled točaka na mapi

Slika 11.14 Izgled iz igračeve perspektive



Slika 11.15 Izgled igre



Slika 11.16 Izgled igre sa druge strane mape

12. Zaključak

Unreal Engine 4 je odličan alat za izradu video igara, sa svojim vizualnim skriptiranjem ljudi koji nemaju velikog znanja iz programiranja mogu napraviti vrlo impresivne igre. Također sam softver je jedan od vizualno ljepših koji trenutno postoje, a ujedno je i besplatan. Sa svojom bogatom bibliotekom svi korisnici mogu vrlo brzo početi istraživati mnogobrojne mogućnosti koje Unreal pruža, a osim toga tu je uvijek i forum na kojemu se može doći do puno novih znanja.

Vizualno skriptiranje je najzanimljiviji dio ovog softvera i izrađeno je vrlo dobro te je jednostavno za shvatiti. Nažalost ono ne pruža zamjenu za standardno programiranje, zbog čega će ljudi koji znaju programirati ipak radije iskoristiti standardni pristup programiranju.

U ovome radu je korišten sustav vizualnog skriptiranja te je uspješno napravljena video igra uz korištenje i mnogih drugih alata koje Unreal pruža. Video igre u konačnici su vrlo popularni izvor zabave koji godišnje zaradi više nego čitava filmska industrija. Upravo zato sve više igara izlazi godišnje, pogotovo indie igara malih kompanija koje koriste Unreal.

13. Literatura

[1] Unreal Engine Documentation Manual, Dostupno na:

https://docs.unrealengine.com/latest/INT/, listopad 2016.

[2] Autodesk Maya overview, Dostupno na:

http://www.autodesk.com/products/maya/overview, listopad 2016.

[3] Pixologic Zbrush, Dostupno na:

http://pixologic.com/zbrush/features/ZBrush4R7/, listopad 2016.

[4] Unreal Engine 4, Project Browser, Dostupno na:

https://docs.unrealengine.com/latest/INT/Engine/Basics/Projects/Browser/, listopad 2016.

[5] Unreal Engine Marketplace, Dostupno na:

https://www.unrealengine.com/marketplace, listopad 2016.

[6] Unreal Engine 4, Starter Content, Dostupno na:

https://docs.unrealengine.com/latest/INT/Engine/Content/Packs/index.html, listopad 2016.

[7] Unreal Engine 4 Introduction to Blueprints, Dostupno na:

http://www.develop-online.net/tutorials/an-introduction-to-unreal-engine-4-blueprints/0210220, listopad 2016.

[8] Unreal Engine 4, Sculpting the landscape, Dostupno na:

https://docs.unrealengine.com/latest/INT/Engine/Landscape/QuickStart/3/, listopad 2016.

[9] Unreal Engine 4, Creating Landscape Materials , Dostupno na:

https://docs.unrealengine.com/latest/INT/Engine/Landscape/QuickStart/4/, listopad 2016.

[10] Unreal Engine 4, Player Start, Dostupno na:

https://docs.unrealengine.com/latest/INT/Engine/Actors/PlayerStart/, listopad 2016.

[11] Unreal Engine 4, Creating Blueprint Classes, Dostupno na:

https://docs.unrealengine.com/latest/INT/Engine/Blueprints/UserGuide/Types/ClassBlueprint/Creation/, listopad 2016.

[12] Unreal Engine 4, Opening Blueprint Classes, Dostupno na:

https://docs.unrealengine.com/latest/INT/Engine/Blueprints/UserGuide/Types/ClassBlueprint/Opening/index.html, listopad 2016.

[13] Unreal Engine 4, Blueprint Variables, Dostupno na:

https://docs.unrealengine.com/latest/INT/Engine/Blueprints/UserGuide/Variables/, listopad 2016.

[14] Unreal Engine 4, Blueprint Arrays, Dostupno na:

https://docs.unrealengine.com/latest/INT/Engine/Blueprints/UserGuide/Arrays/, listopad 2016.

[15] Unreal Engine 4, PlayerController, Dostupno na:

https://docs.unrealengine.com/latest/INT/Gameplay/Framework/Controller/PlayerController/, listopad 2016.

[16] Unreal Engine 4, UMG UI Designer, Dostupno na:

https://docs.unrealengine.com/latest/INT/Engine/UMG/, listopad 2016.

[17] Unreal Engine 4, Animation, Dostupno na:

https://docs.unrealengine.com/latest/INT/Engine/UMG/UserGuide/Animation/index.html, listopad 2016.

[18] Unreal Engine 4, Property Binding, Dostupno na:

https://docs.unrealengine.com/latest/INT/Engine/UMG/UserGuide/PropertyBinding/index.html, listopad 2016.

[19] Unreal Engine 4, Cascade Particle Editor Reference, Dostupno na:

https://docs.unrealengine.com/latest/INT/Engine/Rendering/ParticleSystems/Cascade/, listopad 2016.

[20] Unreal Engine 4, How to Setup Respawns and Checkpoints, Dostupno na:

https://wiki.unrealengine.com/How_to_Setup_Respawns_and_Checkpoints, listopad 2016.

[21] Unreal Engine 4, Blueprint Time Attack, Dostupno na:

https://www.unrealengine.com/blog/new-training-video-blueprint-time-attack, listopad 2016.

Sveučilište Sjever
Sveučilišni centar Varaždin
104. brigade 3, HR–42000 Varaždin

Prijava završnog rada

Definiranje teme završnog rada i povjerenstva

ODJEL: Odjel za multimediju, oblikovanje i primjenu

PRISTUPNIK: Bogdan Kuzminski | MATIČNI BROJ: 5413/601

DATUM: 11.1.2016. | KOLEGIJ: 3D modeliranje

NASLOV RADA: Proces izrade video igre koristeći Unreal Engine 4

MENTOR: Andrija Bernik, dipl. inf. | ZVANJE: Predavač

ČLANOVI POVJERENSTVA:

1. mr.sc. Dragan Matković, v. predavač - predsjednik
2. mr.sc. Vladimir Stanisavljević, v. predavač - član
3. Andrija Bernik, dipl.inf, predavač - mentor
4. Robert Geček, dipl.ing, predavač - zamjenski član
5.

Zadatak završnog rada

BROJ: 465/MM/2016

OPIS:

Izrada 3D video igre predstavlja kreativni proces u kojemu se od početne skice pa do završnog proizvoda uloži mnogo sati rada ovisno o kvaliteti i kompleksnosti projekta. Za izradu se zbog toga koriste tzv. engini koji nam prikazuju sve promjene i kretnje u stvarnom vremenu (eng. real time) i znatno olakšava slaganje scene.

Cilj ovog rada je prikazati kreativni proces koji dovodi do finalne scene koja je zatim spremna za animiranje ili daljnje programiranje za izradu igre. U to ulazi osnovna ideja, rađenje visoko detaljnih modela (eng. high poly), rađenje nisko detaljnih modela (eng. low poly), teksturiranje, izrada kostura za likove i na kraju umetanja svega u sam engine gdje se sve slaže u scenu i nakon toga dodaju detalji kao što su zvuk, fizika, čestice, svjetla. Osim Unreal Engina koristit će se i Autodesk Maya, Zbrush i Adobe Photoshop.

U radu je potrebno prikazati:
- Opis Unreal Engina, Maye, Zbrusha
- Opis korištenih alata
- Opis principa izrade modela
- Objasniti postupak umetanja modela u engine
- Objasniti postupak postavljanja scene
- Izraditi finalnu scenu

ZADATAK URUČEN: 08.06.2016. | POTPIS MENTORA: Bernik

REPUBLIKA HRVATSKA
SVEUČILIŠTE SJEVER

UNIVERSITY NORTH

Sveučilište Sjever

MMI

SVEUČILIŠTE SJEVER

IZJAVA O AUTORSTVU
I
SUGLASNOST ZA JAVNU OBJAVU

Završni/diplomski rad isključivo je autorsko djelo studenta koji je isti izradio te student odgovara za istinitost, izvornost i ispravnost teksta rada. U radu se ne smiju koristiti dijelovi tuđih radova (knjiga, članaka, doktorskih disertacija, magistarskih radova, izvora s interneta, i drugih izvora) bez navođenja izvora i autora navedenih radova. Svi dijelovi tuđih radova moraju biti pravilno navedeni i citirani. Dijelovi tuđih radova koji nisu pravilno citirani, smatraju se plagijatom, odnosno nezakonitim prisvajanjem tuđeg znanstvenog ili stručnoga rada. Sukladno navedenom studenti su dužni potpisati izjavu o autorstvu rada.

Ja, Bogdan Kuzminski *(ime i prezime)* pod punom moralnom, materijalnom i kaznenom odgovornošću, izjavljujem da sam isključivi autor/ica završnog/diplomskog (*obrisati nepotrebno*) rada pod naslovom Proces izrade videa igre koristeći UE4 *(upisati naslov)* te da u navedenom radu nisu na nedozvoljeni način (bez pravilnog citiranja) korišteni dijelovi tuđih radova.

Student/ica:
(upisati ime i prezime)
Kuzminski
(vlastoručni potpis)

Sukladno Zakonu o znanstvenoj djelatnost i visokom obrazovanju završne/diplomske radove sveučilišta su dužna trajno objaviti na javnoj internetskoj bazi sveučilišne knjižnice u sastavu sveučilišta te kopirati u javnu internetsku bazu završnih/diplomskih radova Nacionalne i sveučilišne knjižnice. Završni radovi istovrsnih umjetničkih studija koji se realiziraju kroz umjetnička ostvarenja objavljuju se na odgovarajući način.

Ja, Bogdan Kuzminski *(ime i prezime)* neopozivo izjavljujem da sam suglasan/na s javnom objavom završnog/diplomskog (*obrisati nepotrebno*) rada pod naslovom Proces izrade igre koristeći UE4 *(upisati naslov)* čiji sam autor/ica.

Student/ica:
(upisati ime i prezime)
Kuzminski
(vlastoručni potpis)