SEMINARSKA DOKUMENTACIJA – PERKELE IV

Žiga Simončič

Nejc Smrkolj Koželj

Mitja Maren

1. Hiter pregled

V tej seminarski dokumentaciji bomo opisali kaj vse nam je useplo naredit v zastavljenem času, kako smo to naredili, sproti pa bomo opisali še malo zgodbe in kakšne tehnične podrobnosti.

Dokumentacija je zastavljena tako, da razlaga stvari v takšnem vrstnem redu kot jih doživi uporabnik – torej od zagona igre kronološko naprej.

2. Žanr igre in zgodba

Gra spada v t.i. »akcijske igre igranja vlog« (angl. *ARPG – Action Role Playing Game*). V igri nadzorujemo samo en like (našega junaka), kamera pa mu ves čas sledu, tako da je junak na sredini zaslona. Kamere prosto ne moremo premikati. Junaka kontroliramo predvsem z miško – lev miškin klik za premik, napad in pobiranje orožij.

Zgodba je preprosta – na tem fantazijskem svetu se je spet znašel Diablo in mi ga moramo ubiti. Na poti pa nas seveda poskušajo ustaviti njegove pošasti in demoni.

3. Pregled igre in tehnična implementacija

3.1. Zagon

Ob zagonu igre nas pričaka zelo zanimiv »loading«. Med nalaganjem se naložijo vsi modeli. Tukaj smo malo optimizirali tako, da se vsak model naloži samo enkrat (imamo globalno tabelo z vsemi bufferji vseh različnih objektov). Ko se vse to naloži, začnemo ustvarjati JavaScript objekte. Stvar smo si zastavili zelo objektno-orientirano tako, da je večino logike osebkov v igri zapakirano v JavaScript razrede (npr. Entity, Hero, World, Camera, Item, ...). Nalaganje teh objektov pa je preprosto – nastavimo jim vse potrebne atribute in podamo referenco na vse potrebne bufferje, ki smo jih naložili predhodno. Zatem se pojaji glavni meni.

3.2. Glavni meni

V glavnem meniju je v ozadju slika iz igre Diablo 3, na voljo pa imamo 2 gumba. Vsi meniji so narejeni z uporabo HTML elementov, ki se skrijavo/prikazujejo s pomočjo JavaScripta (to upravlja razred UI). Drugi gumb nam na akcijo hover samo prikaže kratka navodila, s klikom na prvega pa dejansko začnemo z igro.

3.3. Igra

Igro večinoma igramo s klikanjem miške.

Izbiranje objektov je realizirano z vmesnim framebufferjem – vsak objekt ima svoj unikaten ID, ta ID je nato pretvorjen (lahko bi rekli »hashiran«) v unikatno RGB barvo s katero nato narišemo ta objekt v vmesni framebuffer. Na miškin »hover« preverimo kakšne barve je piksel pod miško in RGB barvo tega piksla pretvorimo nazaj v ID, z katerga potem zlahka izvemo nad katerim objektom je miška.

Klik na teren – s klikom na teren premaknemo junaka. Ob kliku najprej pogledamo, če se tam nahaja kak objekt. Če ni nobenega potem z »RayCastingom« dobimo koordinate točke pritiska v svetu (na ravnini). Junaku tja postavimo waypoint.

Tudi pošasti imajo waypointe – ti so nevidni. Ko zagledajo junaka (imajo svoj »viewRange«), se jim nastavi waypoint na junaka. Vsi premikajoči objekti v igri se vsak frame poskusijo premakniti proti waypointu.

Za »pathfinding« ne uporabljamo A*, ampak imamo svoj »domač« algoritem, ki za naše potrebe deluje v redu in menimo, da je veliko bolj optimalen kot A*.

Implementiran je tudi collision. V tem seminarju nismo uporabljali nobenih dodatnih knjižnic (razen glMatrix), tako da smo vse spisali sami. To je tudi razlog, da imamo veliko funkcij in različnih načinov preverjanja collisiona zakomentiranega, saj so različne variante različno dobro delovale. Na koncu smo se odločili za mešan collision. Za premikajoče objekte se uproablja krog, za statične (npr. stena) pa se uporablja kvadrat. Za vsak objekt se »collision box« izračuna ob nalaganju objekta – povprečje med minimalno in maksimalno točko po vsaki koordinati.

Za junaka preventivno preverjamo še y koordinato na kateri stoji. To je narejeno s še enim frame bufferjem, ki ima kamero postavljeno navpično navzdol po y osi, tik nad junakom. S tem pogledamo, kateri objekt je tik pod junakom in če ni nobenega, junaku ne pustimo da bi se premaknil. Tako tudi računamo y koordinato za premikanje po stopnicah vseh premikajočih objektov.

Implementiran je tudi »invenotry« sistem. Pošasti ob smrti za sabo pustijo naključno zgenerirano orožje. Imamo štiri redkosti orožij, vsaka redkost pa ima interval naključnih atributov, ki se lahko zgenerirajo na orožju. Ta orožja lahko s klikom poberemo in nato vzememo v roke v »inventory« meniju.

Bojevanje je preprosto – s klikanjem na pošast izvajao »basic attack«. V kodi je basic attack implementiran kot »Ability« (sposobnost) s svojim cooldownom, rangom itd. Prav tako imamo na voljo še dve druge sposobnosti – 360 slash (oz. »whirlwind«, ima tudi animacijo) in »Heal«, ki nas pozdravi. Ta dva abilitija ima samo junak, pošasti imajo pa samo navaden napad.

Z navadnim napadom imamo možnost kritičnega udarca – koliko je možnosti za tak udarec nam pove »critical chance«.

Če umremo, se luč, ki je nad našim junakm postopno ugašuje, dokler ne postane čisto temno. Nato nastopi »Game over« zaslon. To dosežemo s s preminjanjem t.i. »attenuation« faktorja, ki ga podamo fragment shaderju za izračun svetlosti glede na razdaljo od izvora luči. Nastavljanje tega faktorja se dogaja tudi med bojevanjem z Diablom.

3.4. UI v igri

Uporabniški vmesnik v igri je sestvljen iz sledečih elementov: »boben« za življenje, »boben« za fury in prostora za gumbe in abilitije. Imamo 4 gumbe: gumb za odprtje inventorija, gumb za odprtje »character« zaslona, kjer vidimo podrobne podatke o našem junaku, gumb za pomoč (bolj podrobno razložene mehanike v igri) in gumb za meni. V temu meniju imamo gumb »Exit«, ki nas pelje nazaj na glavni meni in gumb za kupovanje t.i. »Valor pointov«.

Vse te menije posodablja razred UI s pomočjo innerHTML, podatke pa dobi iz objekta Hero.

3.5. Zvok in glasba

Implementirali so tudi zvok in glasbo. Imamo kar veliko različnih posnetkov in efektov, ki so pobrani iz različnih iger. Zvoki so implementirani kot <audio> značke v HTML, ki jih nato razred UI predvaja, ustavlja in nastavlja glasnost.

Tukaj so navedeni vsi zvoki:

- Vsak klik na gumb sproži zvočni efekt klika.
- Zamenjava orožja sproži zvočni efekt
- Ko začnemo igrati se vklopi ambientna glasba
- Vsak udarec pošasti se zaigra naključen zvok udarca
- Vsakič, ko nas kdo rani, se zaigra zvok poškodbe
- Ob pobiranju orožja se zaigra zvok za pobiranje
- Ko pošasti pustijo orožje, se zaigra zvok – 2 različna zvoka glede na redksot orožja
- Ko srečamo Diabla se ambientna glasba zaključi in začne igrati »bossfight« glasba
- Ko nas Diablo zagleda nam pove nekaj lepega
- Mi mu odgovrimo nazaj
- Med bojem nam tudi kaj naključno reče
- Če umre se začne vrteti »Victory« glasba
- Če umremo mi, se ambientna glasba ustavi in se zavrti »Game over« glasba
- V primeru, da smo v trenutni igri umrli in srečali Diabla, nam Diablo ob smrti nekaj reče
- + še nekaj endgame zvokov

3.6. Luči

V igri imamo samo eno luč – nad junakom. To je point light z nekim »attenuation« faktorjem.

Implementirano imamo tudi directional light, ki bi svetila na junaka, ampak je trenutno izklopljena, saj je junak že dovolj osvetljen.

Ambientna svetloba je tudi implementirana, a je postavljena na barvo 0.

3.7. Objekti in teksture

V ekipi nimamo oblikovalca, tako da smo vse objekte dobili iz interneta. Težko je najti modele, ki bodo popolnoma tematsko ustrezali igri. Trenutno smo kar zadovoljni, le moti nas da so nekateri objekti v obliki T in je problem pri računanju collisiona.

Teksture so tudi iz interneta (v glavnem so prišle z objekti), nekatere pa smo sami mapirali na ravnino, ki predstavlja svet.

Trenutni objekti:

- Ironman
- Feral Ghoul
- Pumpkin
- Frog
- Dead Space monster
- Diablo
- Crate
- Trunk
- Wall
- Stair
- World_plane
- Sword model

Ko bomo igro delali v Unitiju, bomo potrebovali vse zgoraj našteto in še več iz vsake kategorije. Po vsej verjetnosti bomo vse objekte zamenjali z novimi iz Unity Stora (tistimi ki so zastonj in imajo zraven teksture in po možnosti še animacije).