Elektrotehnički fakultet Univerziteta u Beogradu
Katedra za računarsku tehniku i informatiku

Razvoj 3D grafike i korisničkog interfejsa didaktičke akcione igre u XNA okruženju

završni rad osnovnih akademskih studija

Mentor

dr Igor Tartalja

Student

Milan Micić

152/2007

Beograd, 2011.

Apstrakt

U radu se opisuje didaktička akciona računarska igra sa fokusom na njeno grafičko okuženje. “The Janitor – cleaning up the mess” je trodimenzionalna igra sa pogledom iz trećeg lica, razvijana za Windows platformu uz pomoć programskog jezika C# i XNA – dodatnog okruženja za razvoj igara. Sama ideja igre je da igrač vodi domara škole kroz razne avanture, pri čemu su avanture tako osmišljene da skreću pažnju igraču na probleme u polju ekologije i navode igrača da rešava te probleme. Pored toga, na kraju priče igrač prolazi kroz kviz čime se upotpunjuje edukativni karakter igre.

Deo razvoja igre u pomenutim tehnologijama koji je opisan u ovom radu, sastoji se iz nekoliko elemenata. Fokus rada biće na prezentaciji razvijenog grafičkog korisničkog interfejsa uz obradu 2D objekata kao i na nadogradnji i prilagođenju postojeće grafičke mašine sa crtanjem i animacijom 3D modela. Pored toga biće izložena i rešenja ostalih načina komunikacije sa korisnicima – pre svega obrada ulaza i zvučnih efekata u igri. Delovi igre vezani za fizičku mašinu i veštačku inteligeniju nisu prednet ovog rada.

Ključne reči: 3D akciona igra, korisnički interfejs, grafička mašina, XNA.

Sadržaj

1. Uvod

Razvoj igara je jedna od mnogobrojnih grana razvoja računarskog softvera. Prve video igre vuku korene još iz šezdesetih godina 20. veka. Iako su tada razvijane po nekolko godina namenski za "*mainframe*" računare i iako nisu bile dostupne javnosti, predstavljale su prve korake u razvoju interesantnih grafičkih okruženja i veštačke inteligencije [1]. Nakon toga, već sedamdesetih godina, pojavile su se komercijalne video igre koje od tada pa do dana današnjeg, kako za konzole tako i za personalne računare, diktiraju tempo razvoja softverskih i hardverskih komponenti za udobnije korišćenje igara, i njihov brži razvoj. Između pokretačke snage razvoja tehnologija koju računarske igre nose i njihovog postojanja zarad zabave, često se potencira i njihov edukativni smisao. Poseban žanr igara GBL (*Game Based Learning* [2]) se bavi aplikacijama koje se razvijaju sa ciljem povezivanja željene materije i toka igre, obraćajući pažnju na mogućnost primene stečenih znanja u realnom svetu.

Imagine Cup je tehnološko takmičenje svetskog nivoa koje Microsoft kompanija organizuje već devet godina. Tema takmičenja je korišćenje novih tehnologija u rešavanju najvećih svetskih problema današnjice. Problemi su ustanovljeni od strane Ujedinjenih Nacija a obuhvataju opšteprihvaćena ljudska prava [3]. Takmičenje se sastoji iz nekoliko kategorija od kojih su neke razvoj softvera, hardvera, digitalnih medija kao i kategorija razvoja igara.

Upravo je ta kategorija predstavljala motivaciju za razvoj edukativne didaktičke igre koja bi svojim igračima uspela da podigne svest o svetskim ekološkim problemima. Želja je da se igraču na specifične načine skrene pažnja na aktuelne probleme pri čemu mu se pruža prilika da kroz avanturu, akciju i šalu učestvuje u njihovom rešavanju. Pored toga, igrač u toku igre osvaja korisne informacije vezane za pomenute teme, koje na kraju igre može iskoristiti u proveri svog znanja kroz kviz.

Ideja takmičenja je i razvoj projekta u timskom okruženju, a uzimajući u obzir postojanje mnogih disciplina kojima se članovi tima mogu baviti u toku stvaranja igre, tema ovog rada neće biti razvoj celokupne igre. Fokus će biti na predstavljanju razvoja grafičkog korisničkog interfejsa, nadogradnje grafičke mašine, kao i načina komunikacije sa igračem, dok će razvoj fizičke mašine i veštačke inteligencije biti tema drugih radova.

U radu će, najpre, pored uslova učešća na takmičenju biti opisana ideja i tematika igre kao i specifični problemi kojima će se posvetiti pažnja. Zatim će biti predstavljeno razvojno okruženje XNA 4.0 bazirano na Microsoft-ovoj .NET 4.0 platformi, sa grafičkom mašinom koju ono nosi kao i korišćena dodatna grafička mašina koja je omogućila lakši rad sa animacijama. Nakon opisa korišćenih tehnologija i funkcionalnih specifikacija igre, biće predstavljena rešavanja konkretnih problema.

2. Problem

Neki od milenijumskih ciljeva Ujedinjenih Nacija, koji su postavljeni kao ciljevi na takmičenju Imagine Cup su: smanjenje siromaštva i gladovanja, pravo na osnovno obrazovanje, pravo na polnu ravnopravnost, smanjenje smrtnosti dece, zaštita trudnica, borba protiv zaraznih bolesti, očuvanje životne sredine kao i unapređivanje globalnog partnerstva za razvoj. Pored toga, igre koje učestvuju na takmičenju u kategoriji igara za Windows/Xbox, moraju da budu bez komunikacije u okviru mreže i moraju da zadovoljavaju „E“ kriterijum organizacije ESRB (*Entertainment Software Rating Board*) – što znači da su pogodne za osobe starije od šest godina [4].

Za igru uopšte, najznačajnija je atraktivnost kako bi igraču privukla pažnju. Međutim, prilikom projektovanja igre koja je predmet ovog rada se morala pronaći odgovarajuća veza između zabavnog dela igre i tema samog takmičenja. Ideja je bila napraviti trodimenzionalnu igru za igrača sa pogledom iz trećeg lica koja će biti edukativnog karaktera. Igru koja će deci mlađeg uzrasta skretati pažnju na realne probleme u polju ekologije.

Pre svega, kako bi deci bilo pistupačno, priču je potrebno da vodi neko iz realnog sveta, neko blizak ali i zaštitnički nastrojen. Školski domar je osoba koja u ovoj prilici dobija glavnu ulogu u kojoj prepričava deci svoje avanture tako što ih kao igrače kroz njih vodi. U avanturama su se globalni problemi širom sveta preslikali u simbolične izazivače problema u školskom dvorištu. Moraće da se bori za očuvanje čistoće dvorišta i obližnjeg jezera, kao i za održavanje kvaliteta školskog vazduha. A kako bi sve izgledalo atraktivnije, pripadnici druge planete žele da se oslobode svog otpada koji je oživeo i domaru ne predstavlja lakog neprijatelja. Jedna od nagrada koju igrač dobija u toku igre je lista korisnih informacija iz realnog sveta na temu ekoloških problema, koja mu kasnije može značiti u proveri svog znanja.

Specifični problemi čiji će načini rešavanja biti izloženi u nastavku, polaze od obrade ulaza i konrole kamere u okruženju XNA. Kako je u pitanju trodimenzionalna igra, jednostavno i intuitivno kontrolisanje kretanja glavnog lika je od velikog značaja.

Drugi problem predstavlja kreiranje grafičkog korisničkog interfejsa. Pre svega kreiranje glavnog menija igre koji se koristi i za meni u toku pauze u igri, ali koji nalazi ulogu i u kvizu na kraju igre. Međutim, pored ovih delova interfejsa, pažnja će biti posvećenja i detaljima koji sačinjavaju korisnički interfejs u toku igre.

Nakon rada sa 2D objektima u XNA okruženju, biće predstavljen način rada i sa 3D objektima. Kako bi se što više doprinelo prirodnom izgledu karaktera iz igre, posebna pažnja se mora posvetiti animacijama 3D objekata. Okruženje XNA nema standardnu podršku za animacije što je razlog korišćenja i prilagođavanja dodatne grafičke mašine. Prikazaće se mogućnosti korišćenja njenog interfejsa kao i potrebna prilagođavanja.

I na kraju, kako bi igrač imao celokupni multimedijalni doživljaj, biće obrađen problem implementacije zvučnih efekata u igru u pomenutom radnom okruženju.

3. Korišćene tehnologije

XNA Game Studio 4.0 predstavlja dodatni IDE (*Integrated Development Environment*) za *Microsoft*-ovo razvojno okruženje *Visual Studio 2010*. Koristi se za kreiranje igara za *Windows Phone*, konzolu *Xbox 360*, kao i za računare bazirane na *Windows* operativnom sistemu. Sama skraćenica okruženja XNA je sa početkom razvoja 2004. godine, nastala na osnovu tadašnjeg naziva projekta – “*Xbox New Architecture*”. Međutim, *Xbox 360* je objavljen 2005. godine, nakon čega se skraćenica šaljivo tumači kao “*XNA is Not an Acronym*”, kako okruženje ne predstavlja podršku samo za *Xbox* [5].

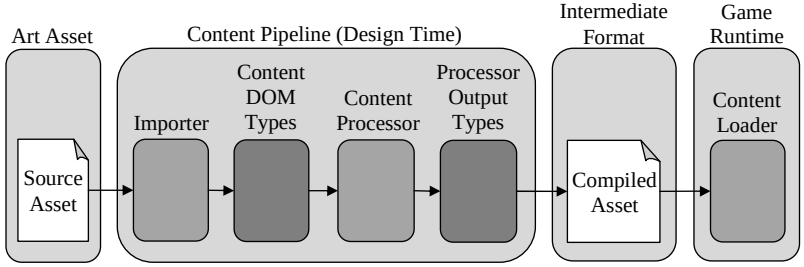
U nastavku će biti prezentovane neke značajne osobine okruženja, koje je korišćeno tokom razvoja igre. Pre svega će se istaći neophodna hardverska i softverska podrška za normalan rad okruženja. Zatim će se proći kroz grafičku mašinu koju nudi okruženje, u vidu protočne obrade sadržaja, obraćajući pažnju na učitavanje i prikazivanje 2D i 3D modela. Posebno će se obratiti pažnja na dodatnu grafičku mašinu i njen interfejs, koja je korišćena tokom razvoja za lakši rad sa animacijama. Nakon toga će biti reči o podrški za obradu ulaza u vidu miša i tastature, ali i izlaza u vidu zvučnih efekata. Na kraju opisa korišćene tehnologije, biće prikazana osnova svake igre koja se ogleda u postojanju njene glavne petlje, koju okruženje nudi.

3.1. Neophodna podrška hardvera i softvera

Operativni sistemi koji u potpunosti podžavaju *XNA Game Studio 4.0* su bilo koje verzije operativnih sistema *Windows XP*, *Windows Vista* i *Windows 7*. Okruženje se zasniva na *Microsoft .NET 4.0* platformi, pa je neophodno korišćenje alata *Visual Studio 2010*. Aplikacije koje se stvaraju pod ovim okruženjem je moguće pisati na bilo kom .NET jeziku, ali su jezici C# i Visual Basic.NET podržani u alatu *XNA Game Studio*. Pored toga, za pokretanje igara na *Windows* platformi, sa hardverske strane je potrebno da grafička kartica podržava najmanje *Shader Model 1.1* kao i *DirectX9.0.c*. Da bi se omogućio razvoj i pokretanje igara za *Windows Phone*, grafička kartica mora da podržava *DirectX10* kao i *WDDM 1.1*, dok je za razvoj igara za *Xbox 360* konzolu, potrebno postojanje hard diska na konzoli [6].

3.2. Protočna obrada sadržaja

XNA Game Studio Content Pipeline predstavlja skup procesa koji se pokreću prilikom prevođenja



Slika 1: Procesi protočne obrade sadržaja

i izvršavanja igre, koja sadrži multimedijalne sadržaje. Procesi u fazi prevođenja (Slika 1) predstavljaju transformaciju multimedijalnog fajla (*Art Asset*) od njegove originalne forme do odgovarajućeg oblika (*Intermidiate Format*), koji će, pri izvršenju, na jednostavan način biti korišćen kroz biblioteke okruženja. Protočna obrada sadržaja je dizajnirana tako da podržava najčešće korišćene formate ulaznih fajlova, ali je moguće i proširiti je, tako da na jednostavan način može da podržava nove formate i nove tipove konverzija [7].

Procesi protočne obrade sadržaja se dele na dva tipa, i to u zavisnosti od trenutka izvršavanja: procese u toku razvoja projekta i procese u toku izvršavanja programa. Razlike između ova dva tipa procesa su predstavljene u Tabeli 1.

Aktivnost	Proces u toku razvoja	Proces u toku izvršavanja
Izvršava se na:	Računaru za razvoj	Uređaju za igru
Pokreće se na:	*Visual Studio*	*Windows*, *Xbox 360* ili *Windows Phone*
Koristi:	*Content Pipeline Class Library*	XNA biblioteku
Distribuira se krajnjem korisniku:	Ne	Da

Tabela 1: Razlika tipova procesa protočne obrade

Procesi u toku razvoja igre se izvršavaju prilikom prevođenja projekta. Tom prilikom se obavlja transformacija multimedijalnog sadržaja iz DCC (*Digital Content Creaton*) formata u objekat koji igra može koristiti tokom izvršavanja.

Prvi korak ka tome se obavlja u uvozniku, koji fajl određenog DCC formata konvertuje u *XNA Game Studio Content Document Object Model* (DOM) format, koji može biti ulazni podatak standardnog procesora protočne obrade sadržaja (*Content Processor*). Takođe, moguće je stvoriti novi uvoznik koji će kao rezultat kreirati željeni format, koji može biti korišćen u specifičnom procesoru protočne obrade sadržaja. Već postojeći uvoznik okruženja stvara objekat DOM formata, koji se sastoji od skupova mreža, kostiju, materijala, tekstura i fontova.

Lista standardnih uvoznika koji postoje u okruženju, kao i formati sadržaja koje podržavaju se nalaze u tabeli (Tabela 2).

Naziv uvoznika	Ulazni formati datoteka	Opis
Autodesk FBX	.fbx	Učitava 3D modele stvorene verzijom 2006.11 FBX izvoznika.
Effect	.fx	Učitava sadržaje formata DirectX efekata.
X File	.x	Učitava 3D model u DirectX formatu – X.

Texture	.bmp, .dds, .dib, .hdr, .jpg, .pfm, .png, .ppm, .tga	Učitava teksturu.
Sprite Font Description	.spritefont	Učitava opis fonta.
MP3 Audio File	.mp3	Učitava audio fajl. Pored toga postoje uvoznici i za formate tipa WAV, WMA.
WMV Video File	.wmv	Učitava video fajl.
XML Content	.xml	Učitava XML sadržaj uz pomoć kog se menjaju vrednosti objekata u toku vremena izvršavanja.

Tabela 2: Standardni importeri i formati koji su podržani

Procesor protočne obrade (*Content Processor*) na ulazu prima samo njemu odgovarajući format objekta. Nakon završetka obrade, odnosno završetka prevođenja, izlaz predstavlja objekat kompaktnog binarnog formata, koji se često naziva međuformat, čija je ekstenzija .xnb. Ovaj format koriste komponente protočne obrade sadržaja u toku izvršavanja programa, i to za dohvatanje i korišćenje multimedijalnih sadžaja. Značajan detalj okruženja je postojanje mogućnosti komprimovanja rezultata prevođenja varijantom LZ algoritma.

U nastavku se nalazi lista postojećih procesora protočne obrade sa opisom (Tabela 2).

Naziv procesa	Izlazni format	Opis
Model	*ModelContent Class*	Parametrizovan procesor koji na izlazu proizvodi 3D objekat. Ulazni parametri omogućavaju neke osnovne radnje sa modelima, nalik postavljanju početne orijentisanosti i faktora skaliranja.
Texture	*TextureContent Class*	Procesor koji stvara 2D objekat. Parametrima je moguće postavljanje veličine i oblika teksture.
Sprite Font Description	*SpriteFontContent*	Datoteku .spritefont, koja predstavlja opdređeni font, pretvara u objekat fonta.
Sprite Font Texture	*SpriteFontContent*	Pretvaranje 2D .bmp teksture u objekat fonta pri čemu parametrom može da se podešava prvi karakter fonta.
No Processing Required	*Object*	Ne obavlja se nikakva odbrada. Koristi se kod sadržaja koji su već spremni za korišćenje u vreme izvršavanja, na primer: .xml dokument sa podacima za definisanje nivoa igre ili postojećih animacija u 3D modelu.

Tabela 3: Postojeći procesori protočne obrade sa odgovarajućim izlaznim formatima

Proces u toku izvršavanja programa – punilac sadržaja (*Content Loader*), omogućava jednostavno i brzo učitavanje svih potrebnih sadržaja kao i njihovo oslobađanje iz memorije.

Nakon učitavanja željenog grafičkog sadržaja moguća je njegova obrada kao i prikazivanje, za šta okruženje nudi svoj korisnički interfejs. U radu sa dvodimenzionalnim teksturama je moguće dohvatanje svih potrebnih podataka vezanih za oblik i veličinu, pamćenje tekstura kao novih datoteka i njihovo iscrtavanje. Iscrtavanje se može obaviti uz podešavanje parametara kao što su: željeni deo teksture, pozicija ekrana, zarotiranost, faktor skaliranja kao i specijalni efekat prilikom iscrtavanja. Prilikom rada sa trodimenzionalnim modelima, interfejs pored jednostavnog načina prikazivanja nudi i olakšan način pozicioniranja kao i pokretanja objekata u 3D svetu. Samo prikazivanje se zasniva na skupu podataka, koje je procesor protočne obrade pripremio. Standardni procesor, kao što je već rečeno, priprema podatke o svim kostima i mrežama, tako da se prikazivanje zasniva na dodeljivanju kamere i odgovarajućeg efekta svakoj od postojećih mreža modela. Interfejs takođe nudi podršku za standardne efekte prilikom iscrtavanja, ali je moguće i kreirati sopstvene efekte pomoću pomenutih .fx datoteka, u jeziku HLSL (*High Level Shader Language*) [8].

Glavni razlog postojanja protočne obrade sadržaja je potreba da funkcionisanje igre bude brzo. Bez njenog postojanja, igra bi morala da koristi multimedijalne sadržaje u originalnoj formi. Tada bi prilikom učitavanja određenog grafičkog objekta zarad njegovog iscrtavanja, igra morala da odredi njegov format i konvertuje podatke u oblik koji se može koristiti. Ovo bi moralo da se odvija u toku izvršavanja igre, čime bi se ono značajno usporilo. Ta pojava se protočnom obradom sadržaja pomera u vreme prevođenja, čime se postiže uključivanje serijalizovanih multimedijalnih sadržaja u izvršni program.

3.3. XNA biblioteka za rad sa animacijama

XNA Animation Component Library je projekat slobodan za korišćenje, čiji je glavni cilj bio kreiranje biblioteke sa potrebnim komponentama za animaciju, pošto standardno okruženje XNA nema takvu podršku [9]. Ta biblioteka omogućava jednostavno animiranje trodimenzionalnih modela.

Biblioteka se sastoji pre svega od izmenjenog procesora protočne obrade za obradu modela. Pored informacija koje priprema standardni procesor protočne obrade, procesor biblioteke za animacije učitava iz modela i skup podataka o animacijama koje postoje. Kontrolu nad podacima modela čije se animacije žele, treba da ima klasa `ModelAnimator`. U njoj se nalaze podaci o svim mrežama, njihovim efektima i kostima čiji se položaji ažuriraju u zavisnosti od animacije koja je pokrenuta. Jedino je potrebno da se animatoru dostavljaju podaci o željenom položaju modela u trodimenzionalnom svetu. Iscrtavanje modela se obavlja pozivom iscrtavanja animatora.

Postavljanje trenutno aktivne animacije modela se vrši pomoću `AnimationController` klase. Za svaku od kreiranih animacija u modelu, potrebno je kreirati instancu kontrolera. Na taj način je moguće pokretanje animacije dodelom kontrolera određenom elementu iz skupa kostiju animatora. Tako je omogućeno i kombinovanje više animacija na jednom modelu u istom trenutku, ukoliko je to potrebno. Biblioteka, kroz svoju implementaciju kostiju modela, nudi i podršku za balansiran prelaz sa jedne na drugu animaciju. Kontroler animacije, takođe, sadrži i

podatke o ukupnoj i protekloj dužini trajanja animacije, faktor ubrzanja kao i mogućnost signalizacije kraja animacije putem događaja koje nudi C#.

3.4. XNA podrška ulaza i izlaza

Standardna podrška za ulazne podatke igrice, koju okruženje XNA pruža, se ogleda u mogućnostima unošenja podataka pomoću tastature, miša i igračkog upravljača za konzolu. Za svaki od tih uređaja postoji interfejs predstavljen odgovarajućom klasom i to respektivno: `Keyboard`, `Mouse` i `GamePad`. Moguće je pristupiti statičkim metodama tih klasa, koje ili dohvataju niz pritisnutih tastera, ili omogućavaju proveru da li je konkretan taster pritisnut ili otpušten. U slučaju miša, moguće je dohvatiti i pomeraj točkića kao i koordinate trenutne pozicije na ekranu, dok u slučaju gejmpada postoji i podrška za proveru konekcije kao i za proveru u kom smeru je džojstik usmeren.

Izlaz igre, pored grafičke predstave, je moguće ostvariti u vidu zvučnih efekata. XNA okruženje nudi mogućnost korišćenja zvučnih datoteka kreiranih pomoću alata XACT (*Microsoft Cross-Platform Audio Creation Tool*). Na taj način je omogućeno doradivanje zvučnih efekata. Pored toga, moguće je i korišćenje uprošćenog interfejsa za zvučne efekte, koji se zasniva na upotrebi datoteka ekstenzija: .mp3, .wav i .wma. U oba slučaja, nakon prolaska datoteka kroz protočno učitavanje, interfejs nudi mogućnosti da se zvučni efekti u toku izvršavanja puštaju, pauziraju i zaustavljaju u željenom trenutku, zatim, da im se podešava jačina i brzina izvođenja, kao i da se dohvati dužina trajanja svakog od njih.

3.5. Glavna petlja igre

Klasa `Game` iz paketa `Xna.Framework` predstavlja implementaciju glavne petlje igre koju čine tri metode - `Initialize()`, `Update()` i `Draw()`. Kreiranje igre se zasniva na redefiniciji ovih metoda. Metoda `Initialize()` u izvedenoj klasi iz klase `Game`, će se izvršiti jednom, pre početka petlje. Ona je odgovorna za postavljanje početnih parametara igre, pre prikazivanja njene prve scene. Nakon toga, u petlji će se pozivati metode `Update()` i `Draw()`, koje su odgovorne za održavanje logike igre i za iscrtavanje grafičkih elemenata svake naredne scene, respektivno.

Moguće je izabrati da korak petlje (promenljiva `TargetElapsedTime`) bude ili fiksne ili promenljive dužine. U prvom slučaju, ona će se pozivati na svakih šezdeset delova sekunde, ili na eksplicitno zadatu dužinu trajanja intervala. Nakon pozivanja metode `Update()` pozvaće se metoda `Draw()`, ukoliko nije vreme da se ponovo pozove `Update()`. Nakon pozivanja metode `Draw()`, petlja će biti u stanju čekanja dok ne dođe vreme za poziv metode `Update()`. Okruženje ima mogućnost da detektuje da se `Update()` metoda izvršava predugo, što signalizira flegom `IsRunningSlowly`. Ukoliko tekuće izvršavanje metode `Update()` traje duže od vrednosti promenljive `TargetElapsedTime`, igra će u tom, kao i u narednim ciklusima propustiti poziv metode `Draw()`, sve dok dužina trajanja metode `Update()` ne bude u okviru dužine trajanja intervala petlje i dok broj pozivanja te metode ne sustigne broj ciklusa petlje. Time se osigurava da će metoda `Update()` biti pozivana približno fiksnom učestanošću. Međutim, učestanost iscrtavanja sa ne garantuje, pošto postoji šansa da se određeni broj poziva metode `Draw()` izgubi, kada se dobija utisak da animacije nisu kontinualne, već “iseckane”.

Pored pozivanja pomenutih metoda izvedene klase iz klase `Game`, u svakom prolasku petlje se pozivaju metode `Update()` i `Draw()` klasa izvedenih iz klase `GameComponent`. Moguće je stvoriti komponente i izvođenjem iz klase `DrawableGameComponent`, za klase kojima nije potrebna metoda `Update()`, već koje samo učitavaju i iscrtavaju grafičke elemente. Klase izvedene iz `GameComponent`, kao i iz `DrawableGameComponent`, moraju pre svega da se registruju u igri kroz parametar metode `Add()`, klase `Components` u paketu `Xna.Framework.Game`.

4. Funkcionalna specifikacija

U ovom poglavlju će biti reči o funkcionalnostima igre uopšteno, sa posebnim fokusom na korisnički interfejs. On se može podeliti na tri celine: interfejs početka igre, u toku igre i kraja igre koji predstavlja kviz znanja.

Na samom početku igre, interfejs je intuitivan (Slika 2). Glavni meni se sastoji od tri stavke od kojih je prva početak igre, druga test znanja i treća izlazak iz igre.

Slika 2: Glavni meni

Ako pokrenemo igru, nakon učitavanja, pred nama će se u školskom dvorištu pojaviti domar sa svojom metlom – u ovom slučaju sredstvom za borbu protiv neprijatelja iz vanzemaljskog broda (Slika 3). Pogled je usmeren ka domaru, pri čemu posmatrač (odnosno kamera) može da se kreće po površi polusfere sa centrom u telu domara. Pored toga, interfejs sadrži i tri tipa pokazivača. U gornjem levom uglu nalazi se radar koji ima ulogu pomagala za kretanje i to tako što usmerava ka određenom cilju ili najbližim neprijateljima. Drugi tip su brojevi, predstavljeni žutom i crvenom bojom u donjem levom uglu ekrana. Žuti broj predstavlja broj sakupljenih "žetona energije" koji povećavaju specijalnu moć domara za brzo kretanje, dok je crveni broj sakupljenih "žetona informacija" koji otvaraju poruke sa korisnim podacima za kasnije rešavanje kviza. Oba tipa žetona se mogu sakupiti nakon eliminisanja neprijatelja. I poslednji tip pokazivača su skale. U gornjem desnom uglu ekrana se nalaze skale za svaku od tri tipa zagađenosti protiv kojih se igrač bori – zagađenost zemlje, vode i vazduha. Igrač u svakom trenutku mora obraćati pažnju na nivo skala kako bi ostao u igri. Poslednja skala, postavljena u donji levi ugao, predstavlja mogućnost korišćenja specijalne moći domara prilikom očuvanja školskog dvorišta – veoma brzog kretanja, koja se povećava sakupljanjem novčića.

Slika 3: Početak igre

Prvi nivo predstavlja borbu zarad očuvanja zemljišta. Neprijatelji u obliku malih ali pokretnih kanti za smeće, nakon izlaska iz vanzemaljske letelice, u šetnji i trku rasipaju smeće. Mogu se primetiti u dvorištu i na radaru (Slika 4), a igrač mora biti spretan i pažljiv kako bi na vreme stigao da ukloni nered i njegove uzročnike. U suprotnom, njihov broj postaje preveliki i zelena skala zagađenosti će stići do

Slika 4: Odbrana školskog zemljišta

maksimuma nakon čega igrač gubi partiju i može da pokuša ponovo, ispočetka.

Kako bi igrač pokazao da je speman da nastavi svoj zadatak, potrebno je da dokaže da može da se suprotstavi i težem užročniku zagađenja ovakve vrste – većem i spretnijem neprijatelju (Slika 5.)

Slika 5: Džinovski neprijatelj

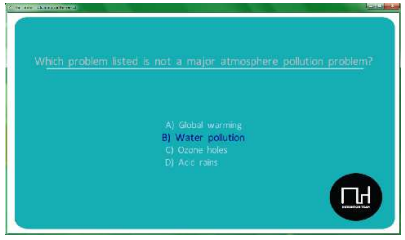
Nakon toga, sledeći nivoi odbrane školskog dvorišta donose sa sobom i nove neprijatelje. Potrebno je zaštititi obližnje školsko jezero, ka kome se kreće mnoštvo neprijateljskih naftnih burića (Slika 6). Oni se, ukoliko su neometani, kreću ka jezeru pronalazeći najbližu putanju ali i beže od igrača ukoliko im se približi. Sada se mora obratiti pažnja na dve skale zagađenja. Pored neprijateljskih kanti, i burići zagađuju zemljište periodičnim ispuštanjem naftnih mrlja, ali im je

Slika 6: Odbrana jezera

Slika 7: Zaštita vazduha

prvenstveni cilj dolazak i uskakanje u vodu čime utiču na povećanje nivoa druge skale.

I na kraju dolazi najteži zadatak – poslednja i najkomplikovanija vrsta neprijatelja, koja ima cilj da uklanjajući obližnju šumu školskog dvorišta, obezbedi prostor za novi otpad sa svoje planete. Samim tim, igrač mora stati u odbranu školskog vazduha, čiju zagađenost predstavlja treća, siva skala. U ovom trenutku igrač mora biti oprezan pošto su neprijatelji istrajni. Ukoliko im se igrač približi, posle određenog vremena uzimanja zaleta, neprijatelji će se ustremiti na njega i ukoliko ga udare, na trenutak će onemogućiti njegovo kretanje (Slika 7). Od takve situacije se može zaštititi skokom.



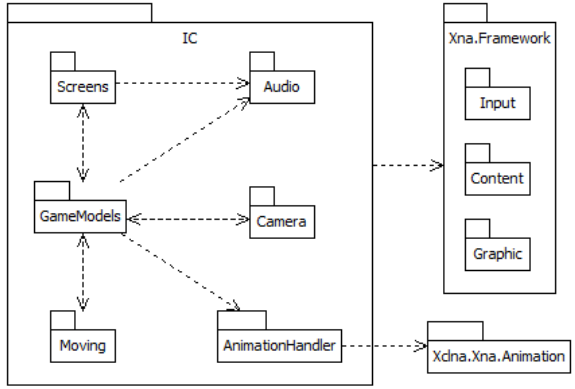
Slika 8: Test znanja

Nakon prevazilaženja svih problema sa neprijateljski nastrojenim otpadom, najistrajniji dobijaju šansu da provere svoje znanje u kvizu (Slika 8). Za prikazivanje pitanja kviza, kao i za mogućnost odabiranja odgovora, iskorišćen je princip po kome je kreiran meni igre o kome će kasnije biti reči.

5. Projekat softvera

Projektovana igra se sastoji od velikog broja paketa, od kojih su u nastavku prikazani oni koji su od interesa za ovaj rad (Slika 9). U narednom delu će, pored načina korišćenja, biti prikazane i glavne karakteristike nekih od njih.

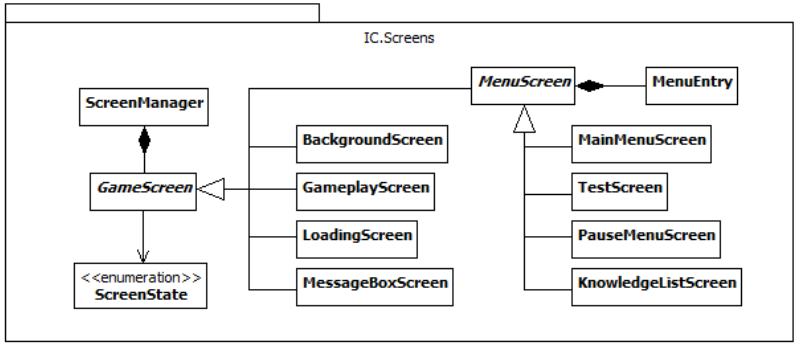
Paket sa nazivom `Xna.Framefork`, predstavlja glavni paket koje XNA okruženje nudi. Interfejs, koji je korišćen tokom rada, se nalazi u njemu i u paketima koje on sadrži. Paket `Xna.Framework.Input` sadrži podršku za rad sa ulaznim uređajima, zatim, paket `Xna.Framework.Content` omogućava učitavanja multimedijalnih objekata kao i njihovo oslobađanje iz memorije i paket `Xna.Framework.Graphic` nudi podršku za korišćenje i obradu grafičkih sadržaja. Mogućnosti koje pomenuti paketi nude, kao i mogućnosti paketa `Xclna.Xna.Animation`, koji predstavlja interfejs za rad sa animacijama, su opisane u trećem poglavlju rada.



Slika 9: Dijagram odabranih paketa projekta

Paket sa nazivom `IC` predstavlja glavni paket projekta. Centralno mesto unutar njega, zauzima paket `GameModels` koji sadrži hijerarhijsku strukturu svih 2D i 3D modela. U njemu se nalazi logika svakog od modela, kao i načini iscrtavanja 2D objekata, pošto se 3D modeli iscrtavaju korišćenjem paketa za rad sa animacijama. Članovi ovog paketa, koji učestvuju u korisničkom interfejsu vidljivom u toku igre, su klase koje predstavlju radar i skale zagađenosti, odnosno, indikatore specijalne moći domara.

Pored dela vidljivog u toku igre, grafički korisnički interfejs se sastoji od ekrana pre početka i nakon kraja igre, čija je podrška ostvarena kroz paket `Screens` (Slika 10). Glavnu ulogu u tom paketu ima klasa `ScreenManager`. Ona je izvedena iz klase `DrowableGameComponent` paketa `Xna.Framework`, i kako takva, njena metoda iscrtavanja će se pozivati svakim prolaskom kroz glavnu petlju igre. Zadužena je za kontrolu prikazivanih ekrana, pa samim tim i njihovog postepenog smenjivanja, pri čemu pruža podršku za dodavanje novih i izbacivanje starih ekrana.



Slika 10: Dijagram klasa paketa za korisnički interfejs

Ekrani, predstavljeni apstraktnom klasom `GameScreen`, na osnovu svog stanja – `ScreenState`, bivaju prikazivani i to po redosledu koji se definiše kod `ScreenManager`-a. Različiti koraci prilikom rada sa ekranima, koji se ogledaju kroz način iscrtavanja, obradu ulaznih podataka i obaveštavanje `ScreenManager`-a o sledećem aktivnom ekranu prilikom napuštanja tekućeg ekrana, su ostvareni korišćenjem projektnog uzorka – *šablonski metod*. Funkcionisanje svakog ekrana, se u osnovi nalazi u klasi `GameScreen`, dok su konkretni koraci ostvareni kroz potklase koje odgovaraju različitim tipovima ekrana:

- `BackgroundScreen` je ekran koji će se icrtavati u pozadini nekog od ostalih ekrana;
- `GameplayScreen` predstavlja ekran čija je glavna uloga komunikacija sa paketom `IC.GameModels` zarad pokretanja igre;
- `LoadingScreen` predstavlja grafički prikaz koji se pojavljuje ukoliko učitavanje komponenti sledećeg ekrana dugo traje. To učitavanje obavlja punilac sadržaja, kako bi podaci o korišćenim objektima bili dostupni tokom igre;
- `MessageBoxScreen` iscrtava obaveštajne poruke na postojećem ekranu ili poruke koje zahtevaju potvrdu, kao što je slučaj u trenutku kada korisnik želi da napusti aplikaciju;
- `MenuScreen` je apstraktna klasa koja predstavlja ekrane koji sadrže instance klase `MenuEntry` – stavke menija koje je moguće odabrati. Na taj način su ostvareni:
  - ekran glavnog menija – `MainMenuScreen`,

  - test na kraju igre – `TestScreen`, koji ima mogućnost odabiranja jednog od četiri ponuđena odgovora,
  - meni u toku pauze igre – `PauseMenuScreen`,
  - provera poznavanja informacija o ekološkim problemima sakupljenih u toku igre – `KnowledgeListScreen`, do kojih je moguće doći iz menija u toku pauze igre.

Podrška zvučnih efekata se nalazi u paketu `IC.Audio`. Kreiran je po principu projektnog uzorka – *muva*. Apstraktnu klasu `SoundCollection` implementiraju konkretne kolekcije zvukova i to posebna za svaku od komponenati igre koje proizvode zvukove. Učitavanje svih zvukova je potrebno obaviti pre početka igre, puniocem sadržaja, kako bi kasnije bilo koji od njih mogao da se koristi. Međutim, kreiranje njihovih instanci je moguće obavljati tek prilikom prve potrebe za određenim zvukom. Takvim instancama je moguće podešavati jačinu i ostale parametre, koji su ranije opisani u radu.

Paket okruženja `Xna.Framework.Input`, kao što je ranije pominjano, predstavlja podršku okruženja za rad sa ulaznim uređajima. Događaji na tastaturi i mišu uzrokuju promene u igri koje utiču na kretanje glavnog igrača, pa samim tim i na animacije koje prate njegov model, dok događaji na mišu dodatno utiču i na kretanje kamere oko glavnog igrača. U paketu `IC.Camera` se nalazi definisana logika kretanja kamere po sferi određenog radijusa čiji je centar u telu domara, usled pomeranja miša. Podrška koju okruženje nudi se ogleda u mogućnostima pozicioniranja oka kamere na određenom mestu i sa određenim pravcem, kao i postavljanjem odgovarajućeg opsega perspektive kao i bliže i dalje ravni odsecanja iscrtavanih modela. Detekcija kolizije kamere i ostalih objekata igre nije predmet ovog rada.

Ulazni podaci koji dolaze sa tastature i utiču na akcije domara, se obrađuje u paketu `IC.AnimationHandler`. U njemu se u zavisnosti od događaja sa tastature, odlučuje koja će animacija biti aktivna, kao što se i obavlja postepeni prelaz sa jedne na drugu animaciju. Podržane su sve animacije koje su kreirane u trodimenzionalnom modelu, kao i njihovo kombinovanje u zavisnosti od potrebe. Pored animacija domara, u ovom paketu se nalazi i podrška za aktiviranje i prelazak sa jedne na drugu animaciju modela neprijateljskih jedinica. Zbog toga je, korišćenje ovog paketa potrebno u paketu zaduženom za logiku veštačke inteligencije, koji nije predmet ovog rada. Takođe, detekcija kolizije glavnog igrača i ostalih objekata, nije predmet ovog rada.

6. Implementacija softvera

Prenosivost igre je ostvarena uz pomoć alata za objavljivanje verzija aplikacije, koji postoji u okviru radnog okruženja *Microsoft Visual Studio 2010*. Na taj način je kreirana instalacija koja, ukoliko je to potrebno, uz uslov postojanja direktne Internet veze, samostalno aktivira preuzimanje sa Interneta i instalaciju dodataka za radno okruženje u vidu .NET 4.0 i XNA 4.0 biblioteka. Nakon toga, instalacija sa prenosivog uređaja na kome se nalazi, instalira biblioteku dodatnog okruženja za rad sa animacijma, a zatim na hard disk domaćina, kopira multimedijalne datoteke u formatu .xnb, koji su potrebni za normalan rad igre. Instalacija omogućava kreiranje prečice ka igri u startnom meniju *Windows* operativnog sistema. Pored toga, instalacija nudi opciju uklanjanja igre sa računara u okviru paketa *Control Panel*, pomoću aplikacije *Add or Remove programs*.

Multimedijalni elementi igre, koji su u potpunosti kreirani od strane dizajnerskog dela tima u programima *3D Studio Max*, kao i *Maya*, u originalnom formatu zauzimaju veličinu od 110MB. Instalacija mora da sadrži sve te elemente koji, poređenja radi, nakon pomenute protočne obrade sadržaja, zauzimaju veličinu od 790MB. Međutim, korišćenjem LZ algoritma prilikom komprimovanja multimedijalnog sadržaja, koje nudi XNA okruženje – što je ranije opisano u poglavlju 3.2, ukupna veličina multimedijalnih sadržaja u instalaciji iznosi oko 320MB, što je značajna ušteda.

Prostor za kreiranje aplikacije u okruženju *Visual Studio*, pored projekta vezanih za dodatno okruženje za rad sa animacijama, ima i projekat koji sadrži sve izvorne multimedijalne sadržaje, kao i projekat koji sadrži izvorne kodove. Izvorni kod je podeljen u 12 paketa koji ukupno sadrže 94 klase, u kojima ima oko 15 hiljada linija koda.

7. Zaključak

Inovativnost svakog projekta, pa i računarskih igara, predstavlja ključan detalj proizvodnje. Taj detalj je i jedan od kriterijuma ocenjivanja projekata na takmičenju Imagine Cup. Opisana igra “The Janitor – cleaning up the mess“ na svoj način uvodi edukaciju na temu ekoloških problema u svet igara. Borba dobra protiv zla će uvek biti atraktivna tema, ali uz realističan lik domara kao prihvatljivog svim generacijama, a pogotovo deci, borba dobija novi oblik.

Projektovana aplikacija je do sada pronašla primenu u učešću na takmičenju o kome je bilo reči. Međutim, najvažniji rezultat razvoja ovakve računarske igre predstavlja dobru osnovu za razvoj bilo kakve druge edukativne igre. Uz odgovarajuću nadogradnju modela i efekata moguće je projektovati komercijalne aplikacije koje bi mogle da pronađu svoje mesto na današnjem tržištu. Ustanovljene su neophodne komponente svake igre, kao i važne tehnike rešavanja problema u toku razvoja igara uopšte, ali i konkretno – u razvojnom okruženju XNA, koje je relativno mlado, ali perspektivno.

Budući razvoj aplikacije bi na osnovu svega do sada pomenutog, mogao da teče u mnogim pravcima. Pre svega, potrebno je ostvariti rešavanje postojećih otvorenih problema. Deo grafičkih komponenti je moguće poboljšati stvaranjem boljih vizuelnih efekata, kako dvodimenzionalnih, tako i trodimezionalnih. Neki od problema u delu logike igre bi bili rešeni boljom organizacijom nivoa igre i testa znanja, ali je putanja daljeg razvoja ostvariva i u pravcu kreiranja novih avantura, sposobosti domara i novih problema kojima će se baviti. Svakako postojeći teren, sastavljen od škole i njenog dvorišta, predstavlja širok prostor za dodavanje novih poučnih avantura. Takva usavršavanja bi bila korisna za planove učešća na sledećem takmičenju Imagine Cup, ali su, naravno, moguća usavršavanja igre i na temu igranja u mreži, što bi bilo korisno za planove komercijalizacije igre.

8. Literatura

[1] Video game development, http://en.wikipedia.org/wiki/Video_game_development

[2] Game based learning, http://en.wikipedia.org/wiki/Game_based_learning

[3] Imagine Cup, http://www.imaginecup.com/CompetitionsContent/2012Theme.aspx

[4] Entertainment Software Rating Board, http://en.wikipedia.org/wiki/Entertainment_Software_Rating_Board

[5] Microsoft XNA, http://en.wikipedia.org/wiki/Microsoft_XNA

[6] Hardwere and Operating System Requirements, XNA Game Studio 4.0, http://msdn.microsoft.com/en-us/library/bb203925.aspx

[7] Adding new Content Types, XNA Game Studio 4.0, http://msdn.microsoft.com/en-us/library/ff827626.aspx

[8] Aaron Reed, *Learning XNA 3.0*, O'REILLY, 2009

[9] XNA Animation Component Librarz, http://animationcomponents.codeplex.com/