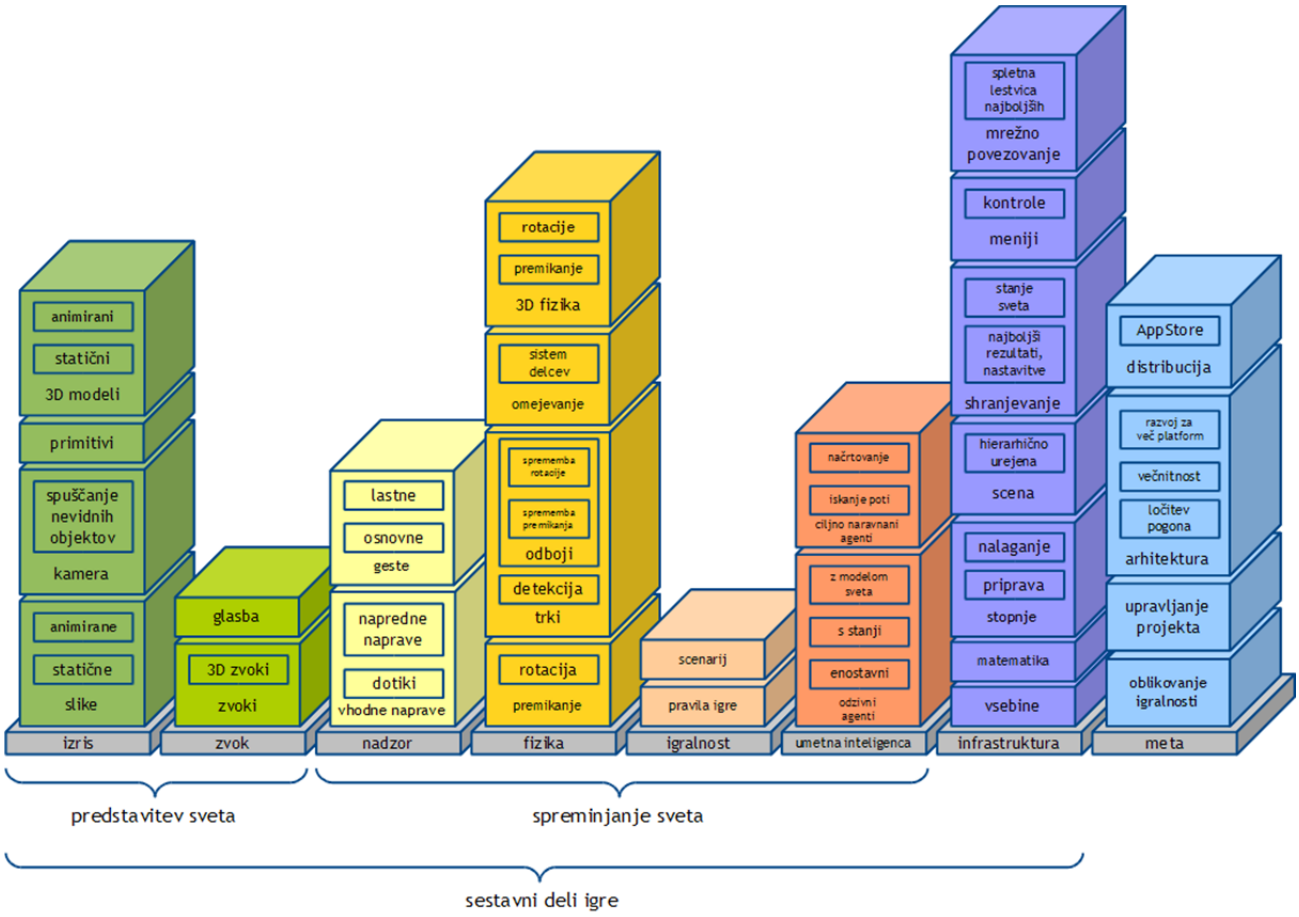
Razvoj računalniških iger
za iPhone in iPad z uporabo ogrodja XNI

Matej Jan

Skripta za praktični del predmeta
Tehnologija iger in navidezna resničnost 2011

dopolnil in uredil Bojan Klemenc

Vsebina


animirani
statični
3D modeli
primitivi
spuščanje nevidnih objektov
kamera
animirane
statične
slike
izris
glasba
3D zvoki
zvoki
zvok
lastne
osnovne
geste
napredne naprave
dotiki
vhodne naprave
nadzor
rotacije
premikanje
3D fizika
sistem delcev
omejevanje
sprememba rotacije
sprememba premikanja
odboji
detekcija
trki
rotacija
premikanje
fizika
scenarij
pravila igre
igralnost
načrtovanje
iskanje poti
cilino naravnani agenti
z modelom sveta
s stanji
enostavni
odzivni agenti
umetna inteligenca
spletna lestvica najboljših
mrežno povezovanje
kontrole
meniji
stanje sveta
najboljši rezultati, nastavitve
shranjevanje
hierarhično urejena
scena
nalaganje
priprava
stopnje
matematika
vsebine
infrastruktura
AppStore
distribucija
razvoj za več platform
večnitnost
ločitev pogona
arhitektura
upravljanje projekta
oblikovanje igralnosti
meta
predstavitev sveta
spreminjanje sveta
sestavni deli igre

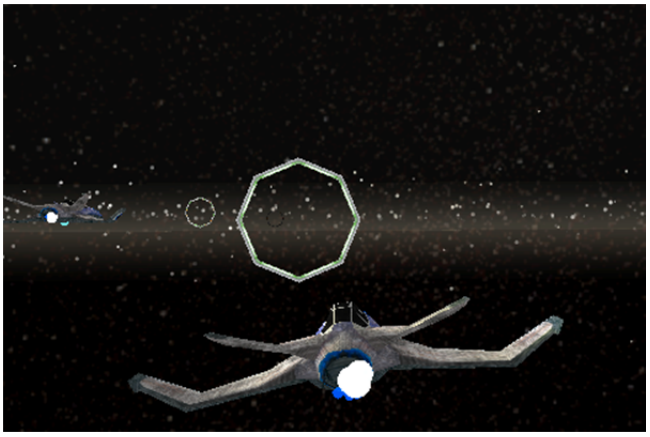
Uvod

Ali kaj imajo skupnega Xbox Live Indie Games, Engadget in XNI

Dokler se ukvarjamo z izdelovanjem iger za hobi, je praktično nepomembno, za katero platformo in s katero tehnologijo to počnemo. Sam sem dolga leta vztrajal na visual basicu, ki mi je bil naravni naslednik basica z zx spectruma.

2006 je Microsoft tudi sam podprl izdelavo iger v .NET okolju. Presedlal sem na XNA in C#, ki je sicer edini uradno podprt programski jezik za to ogrodje. Prepričan sem bil, da je to to in smelo izdelal poslovni načrt, ki bi me iz garažnega programerja ponesel v »indie svet« z izdelavo iger za Xbox Live Indie Games (takrat imenovan še Community Games). Nekaj mesecev pred splovitvijo tega prodajnega kanala pa je začela pot še ena storitev, Applov AppStore. Kot mac-uporabnik, ki se je v okolje windows vračal le zaradi ljubezni do XNAja, se flirtanju s popularnostjo aplikacij za iPhone nisem mogel upreti. Uspeh platforme in podprtost Slovenije (Xbox-a namreč pri nas uradno še vedno ni, kaj šele, da bi bilo omogočeno služiti od prodaje XNA iger na Xboxu) sta me prepričali, da se za začetek osredotočim na iOS.

Ko sem v roke dobil projekt, kjer je bilo potrebno za iPhone razviti igro, ki na zanimiv način uporablja pospeškometer (v povezavi s projiciranjem slike na steno), sem si moral postaviti osnovno arhitekturo za razvoj. Kaj boljšega, kot da si za zgled vzamem XNA. Tako je nastala kopija XNAja v objective-cju, zametek ogrodja XNI. Prototip igre je pristal celo na Engadget.com.



XNA je septembra 2010 doživel precejšnjo nadgradnjo na verzijo 4.0, s podporo za novo platformo Windows Phone 7 ter še izboljšano arhitekturo samega ogrodja. XNI je kopija prav te najnovejše, 4.0 arhitekture in prinaša v objective-c možnost programiranja iger na zelo objektno orientiran način, ki recimo na področju grafike popolnoma skrije proceduralno naravno knjižnice OpenGL.

Tako lahko s semantično skoraj identičnim programom izdelamo igro za iPhone in iPad s pomočjo XNI ter Windows, Xbox 360 in Windows Phone 7 z XNA. Po drugi strani XNI omogoča veliko lažji vstop v razvoj iger za iOS za tiste, ki so se že spoznali z XNA. Če pa si vseeno želimo pogledati, kaj vse se dogaja v ozadju, je to seveda mogoče, saj je XNI odprtokodni projekt.

Tehnologija iger in navidezna resničnost

Razvoj računalniških iger v sklopu predmeta TINR

Predmet Tehnologija iger in navidezna resničnost pokriva zelo razgibano področje izdelovanja iger. Namesto, da bi se lotili vsakega dela posebej in ga obdelali v celoti, bomo najprej spoznali minimalno osnovo vsakega ter tako zgradili podlago, na kateri bo mogoče graditi naprej.

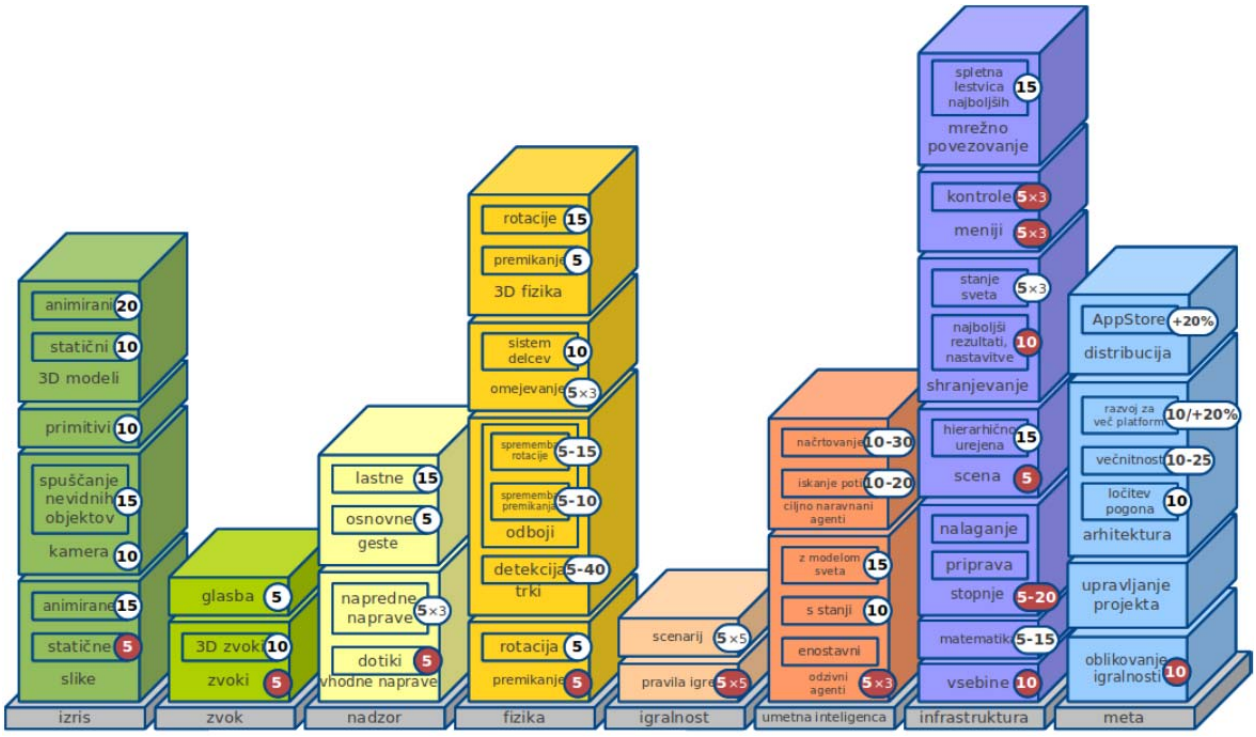
Program je tako razdeljen na dva dela. V prvem bomo poskusili zajeti vse bistvene dele pogona igre (izris, vhod, fiziko, umetno inteligenco, zvok, grafični vmesnik) in z njimi sestaviti prototip igralnosti.

V drugem delu bomo vsako tematiko poglobili z naprednejšimi tehnikami, na vas pa je, da implementirate tiste, ki jih potrebuje za svoj projekt.

Zahteve predmeta

Vaš cilj je, da izdelate zaključen izdelek. Igro, ki naj bi bila pripravljena za objavo. Zato je zelo pomembno, da so ambicije, vsaj kar se obsega tiče, zelo dosegljive.

Za ocenjevanje bo skrbel transparenten sistem točk, ki jih osvojite za implementacijo funkcionalnosti. To boste demonstrirali skozi tedensko oddane posnetke zaslona, v katere na jedrnat način vključite vse na novo razvite vsebine.



Na ta način boste že s sledenjem prvemu delu semestra pokrili skoraj vse zahteve za uspešno opravljen predmet. V drugem delu je na vas in vaših ambicijah, da igro razširite z naprednejšimi elementi. Tako si lahko prav vsak z malce truda in želje po dejanski uporabi naučenega znanja zasluži desetko.

Verjamem namreč, da je najpomembnejše, če se znamo spoznanih tematik skozi prakso naučiti in jih uporabiti ob delu na konkretnem projektu. Od piflanja teorije in formul s faksa verjetno nismo prav veliko odnesli.

Računalniške igre in navidezna resničnost

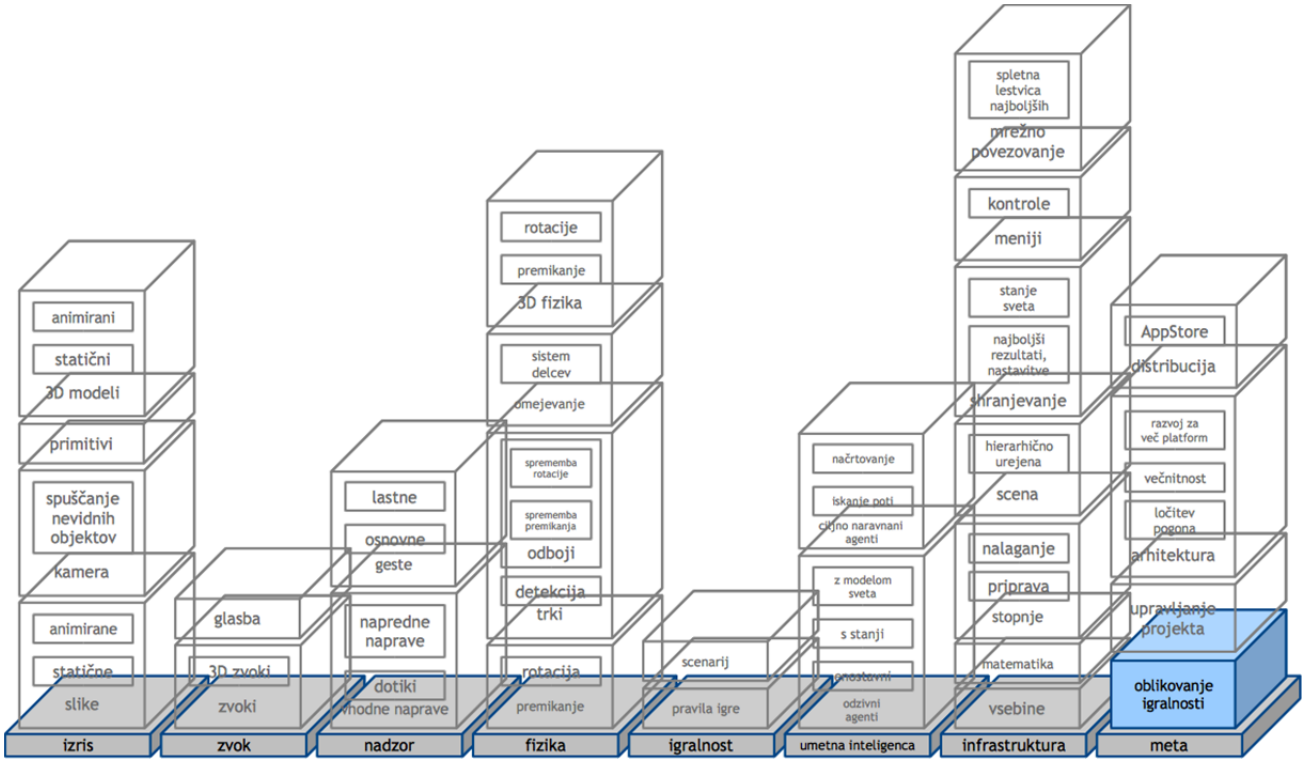
Naziv predmeta je sestavljen iz dveh delov. Tehnologija iger pomeni znanja, ki so potrebna za razvoj računalniških iger. Navidezna resničnost je na drugi strani koncept, ki temelji na zamenjavi sveta okoli nas z navideznim, izmišljenim. Računalniške igre se od običajnih, torej takih, ki smo se jih igrali predvsem kot otroci v resničnosti (igram za odrasle se ponavadi reče športi, da se kdo ne počuti otročjega), razlikujejo v tem, da se zaradi svoje narave izvajajo izključno v navidezni resničnosti, pa naj gre za izmišljena vilinska kraljestva ali samo površino (navidezne) mize, na kateri igramo (posledično virtualni) poker.

Ni pa vsako dogajanje v navidezni resničnosti igra. Sprehajanje po Second Worldu je v osnovi samo raziskovanje sveta, fiktivni svet Matrice je simulacija življenja z edinim namenom zaposliti možgane povezanih, medtem ko njihova telesa služijo kot baterije za robote. Uporaba virtualnosti postane igra šele, ko vanjo vnesemo igralnost. Za to potrebujemo skupek elementov oziroma prvin igralnosti (stvari, ki jih lahko počnemo in pravil, ki nas omejujejo) ter cilj(e). Brez cilja je igra namreč igrača in lahko z njo počnemo karkoli nam omogoča lastna kreativnost. Stremenje k jasnemu cilju, pa naj bo to zadetek čim več golov pri nogometu ali največje število ubitih nasprotnikov v Unreal Tournamentu, šele ustvari igro.

Za konec še eno razlikovanje. Namreč, niso vse igre namenjene zabavi. Resne igre (serious games) se od običajnih ločijo ravno po tem, da cilj ni kratkočasenje, temveč večinoma primarno izobraževanje.

Tehnologija iger torej pokriva vsa ta področja in s tem zajema znanje vseh njih.

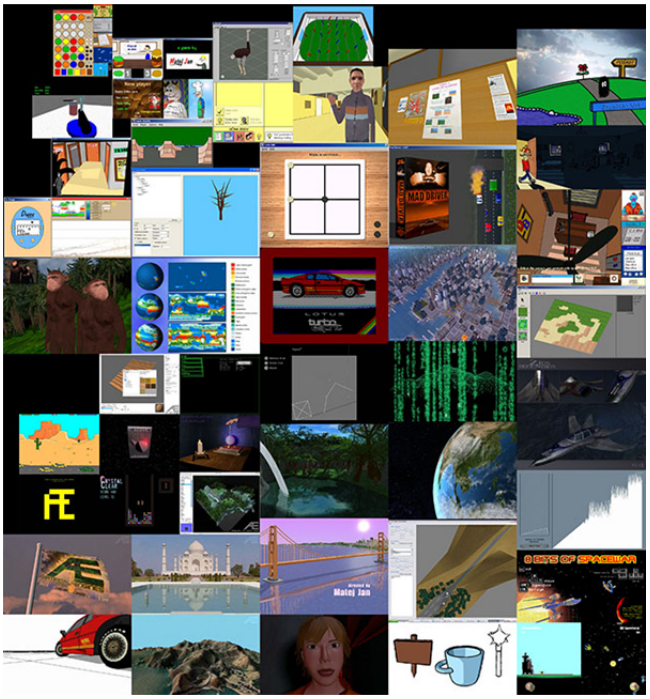
Sklop 1


animirani
statični
3D modeli
primitivi
spuščanje nevidnih objektov
kamera
animirane
statične
slike
izris
glasba
3D zvoki
zvoki
zvok
lastne
osnovne geste
napredne naprave
dotiki
vhodne naprave
nadzor
rotacije
premikanje
3D fizika
sistem delcev
omejevanje
sprememba rotacije
sprememba premikanja
odboji
detekcija trki
rotacija
premikanje
fizika
scenarij
pravila igre
igralnost
načrtovanje
iskanje poti
cilj
agenti
z modelom sveta
s stanji
odzivni agenti
umetna inteligenca
spletna lestvica najboljših
mrežno povezovanje
kontrole
meniji
stanje sveta
najboljši rezultati, nastavitve
shranjevanje
hierarhično urejena
scena
nalaganje
priprava
stopnje
matematika
vsebine
infrastruktura
AppStore
distribucija
razvoj za več platform
večnitnost
ločitev pogona
arhitektura
upravljanje projekta
oblikovanje igralnosti
meta

Od ideje do načrta

Kaj sploh ustvarjati?

Najpomembnejša stvar, ki jo lahko povem nekomu, ki se ravno začne ukvarjati z izdelovanjem iger je to, da naj začne z majhnim in počasi gradi proti večjim idejam. Hkrati je to nasvet, ki do zdaj še nobenemu razvoja iger željnemu navdušencu ni pomagal. To je enostavno stvar, ki se jo moraš naučiti iz svojih izkušenj. Ali bolje rečeno iz svojih n nedokončanih projektov.



Tokrat imam to možnost, da vas prisilim, da začnete z majhnimi projekti. Mogoče vas bom s tem prikrajšal za delček učenja na lastnih napakah, a po drugi strani boste za razliko od mene, ko sem opravljal podoben predmet, na koncu semestra imeli zaključen izdelek, vreden objave.

Elementi igralnosti

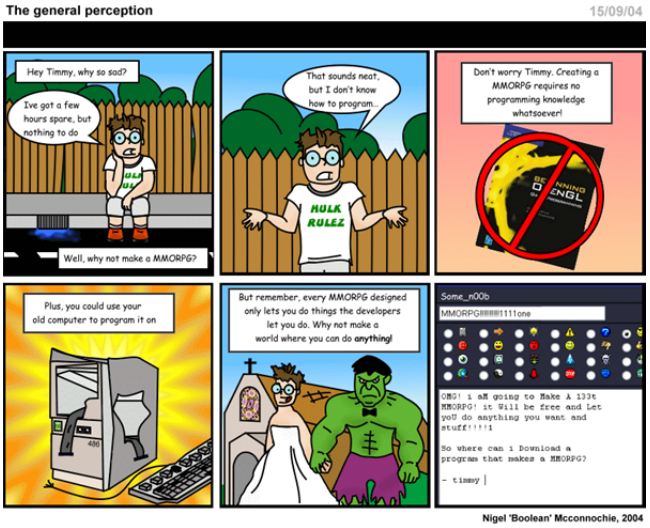
Kako bomo dosegli, da igra ne bo preambiciozna? Z omejitvijo elementov igralnosti.

O definiciji se lahko prerekamo. Zame je element igralnosti prav vsaka stvar, ki sestavlja, spreminja, definira ali omejuje svet igre. Tako jih lahko zelo enostavno identificiramo s pogledom na posnetek zaslona in naštevanjem vseh objektov, ki ga sestavljajo. Primeri so podani v ločenem dokumentu (elementi igralnosti.pdf).

Cilj je torej najti lasten skupek elementov igralnosti, nekje med 5 in 10, ki bodo opisovale vašo igro. Proces tega iskanja pa ni nič drugega kot oblikovanje igralnosti.

Oblikovanje igralnosti

Oblikovanje igralnost (angl. game design) je verjetno kar najbolj všečen del izdelovanja iger, saj v njem fantaziramo o GTAju v svetu MMO, s to razliko, da bo lahko igralec počel, kar bo hotel.



Čeprav gre večina oblikovanja igralnosti res v to smer, da se na že preverjene koncepte samo nalaga izboljšave (na misel mi pride serija Tony Hawk's Pro Skater), je pravi čar dobrega oblikovanja ravno obraten. Ali kot to prebere Leonard Nimoy ob iznajdbi inžinirstva v Civlization IV: "A designer knows he has achieved perfection not when there is nothing left to add, but when there is nothing left to take away," (sicer zapis Antoine de Saint-Exupry-ja, avtorja Malega Princa, o oblikovanju letal).

Ključ je marsikdaj torej ravno obraten in bo še kako pomemben za naše vaje. Kako z namanj možnimi elementi igralnosti še prenesemo sporočilo, ustvarimo občutke ali preprosto omogočimo igranje, ki ga želimo.

Iskanje vzorov

Ni se nujno spomniti nekaj povsem novega. Lahko izdelate popolno kopijo neke starejše igre. »Remake«, kot se temu reče po angleško, je konec koncev med igralci zaželena vrsta igre, saj prinaša nekaj nostalgično privlačnega posodobljeno za današnje čase, pa naj bo to samo zamenjava platforme, izboljšava grafike ali kar oboje skupaj (recimo Monkey Island Special Edition, le ena izmed iger, ki na mojem Steam-računu vztrajno čaka, da bom imel čas zanjo).

Bodite pozorni na letnico originala in izbirajte od 1980 nazaj. Začnete lahko z vašo prvo najljubšo igro in pogledate, kateri so bili njeni (idejni) predhodniki. Poiščite še predhodnike predhodnikov in postopek ponavljajte, dokler ne pridete do nečesa iz sedemdesetih. V pomoč na sprehodu po prajuhi iger so tu Abandonia (dos), World of Spectrum (zx spectrum), Lemon64 (commodore 64), Arcade Museum (avtomati) in nenazadnje najboljša baza vseh iger in razvijalcev MobyGames. Za posnetke igranja pa seveda presrfajte YouTube.

Naloga: oblikovanje igralnosti

Zdaj vam preostane le to, da mi svojo idejo predstavite. Dokumentu, ki opisuje samo bistvo igre na ponavadi ne več kot 1 ali 2 straneh, se reče koncept igre (game-concept document).

Sama vsebina je precej odvisna, komu bi radi predstavili svojo idejo. Tokrat nas zanimajo predvsem konkretni elementi igralnosti in ne tržniška besedila z alinejami o najboljši grafiki in nemogočih obljubah.

Uporabite predlogo (koncept igre - predloga.doc), ki zajema vse smiselne stvari za okvire našega predmeta. To je tudi vsa dokumentacija, ki jo potrebujete za izdelavo prototipa, kar bo vaša naloga v prvem delu predmeta.

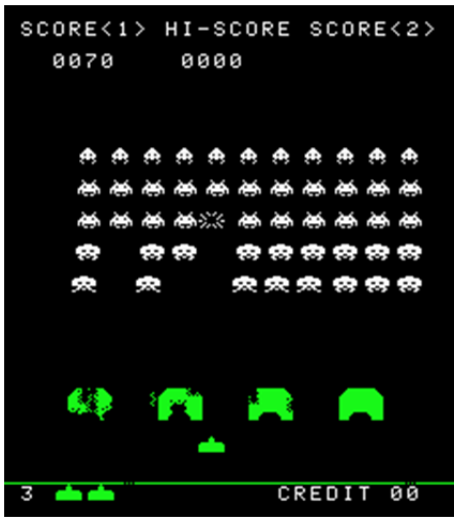
Primeri projektov

Za lažjo predstavo, na kakšne projekte ciljamo, je tu nekaj primerov za navdih.

Najprej si poglejmo retro pristop, pri katerem vzamemo kakšno staro klasiko in izdelamo predelavo.

Osvajalci

Vzor: Space Invaders, Taito, 1978



Streljanje nasprotnikov, ki se počasi premikajo proti igralcu.

Elementi igralnosti:

- ladjica
  - premikanje levo in desno
  - streljanje
- osvajalci
  - skupinsko premikanje
  - streljanje
- barikade
- trki izstrelkov z ladjico, osvajalci in barikadami
- točkovanje
  - štetje nezemljanov
  - življenja igralca
  - najboljši rezultat
- uvodni zaslon

Padalec

Vzor: Sky Diver, Atari, 1978

Igralec skoči iz letala, odpre padalo in poskuša pristati čim bližje cilju. Kasneje kot odpre padalo, več točk dobi.

Elementi igralnosti:

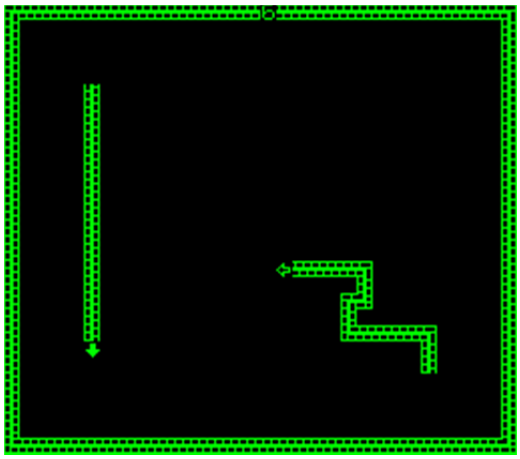
- letalo
  - leti samodejno čez ekran
- padalec
  - skok iz letala, odpiranje padala
  - jadranje levo/desno preko dotikov
- točkovanje
  - meritev časa pred odprtjem padala
  - meritev razdalje od cilja pristanka
  - najboljši rezultat
- uvodni zaslon

Nadgradnje:

- jadranje z nagibanjem
- animiranje padala s sistemom delcev
- izris padala s primitivi
- način za 2 igralca

Kača

Vzor: Blockade, Gremlin, 1976



Igralec vodi kačo po ekranu, ki se s pobiranjem hrane po zaslonu povečuje. Kača se ne sme zadeti v steno ali sama vase.

Elementi igralnosti:

- kača
  - sestavljena iz posameznih delov
  - premikanje pod kotom 90 stopinj
  - daljšanje
- hrana
  - naključno prikazovanje
- stene
- trki s steno in s posameznimi dele kače

Žaba

Vzor: Frogger, Konami, 1982

Z žabo se moramo prebiti čez cesto in vodo do vrha ekrana.

Elementi igralnosti:

- žaba
  - skakanje v štiri smeri

    - zadetek v rob ekrana
- vozila
    - premikanje čez ekran
    - povozijo žabo
- hlodi
    - različnih dolžin
    - različno hitro premikanje čez ekran
- želve
    - različno hitro premikanje čez ekran
    - različno velike skupine
    - potapljanje v vodo
- premikanje žabe, ko je na želvi ali hlodu
- točkovanje
    - prihod na vrh stopnje
        - 5 ciljnih točk
    - odštevanje časa
    - življenja
    - najboljši rezultat
- uvodni zaslon

Nadgradnje:

- več stopenj z različnimi elementi
- krokodili, kače, vidre
- hrošči
- žabica

Dirkanje

Vzor: Road Fighter, Konami, 1984

Dirkanje po cesti iz ptičje perspektive s preprostim vzporednim premikanjem levo in desno.

Elementi igralnosti:

- igralčev avtomobil
  - zavijanje levo/desno preko dotikov
  - avtomatsko pospeševanje
- večpasovna cesta brez zavojev
  - trk z robom ceste
- vozila v prometu
  - trki med vozili
- točkovanje
  - meritev in pokazatelj prevožene proge
  - doseg konca ceste
  - najboljši rezultat
- demo zaslon

Nadgradnje:

- zavijanje z nagibanjem (pospeškometer)
- zaviranje z dotikom
- dvostopenjski menjalnik
- zavijanje ceste
- spreminjanje števila pasov
- vnaprej definirane stopnje
- stopnje z urejeno okolico
- drugi dirkači z agresivnejšo umetno inteligenco
  - pokazatelj mesta v dirki
- zdrsavanje (drift)
- poraba bencina
  - pobiranje kantic
- lestvica najboljših
  - vpis imena
- ...

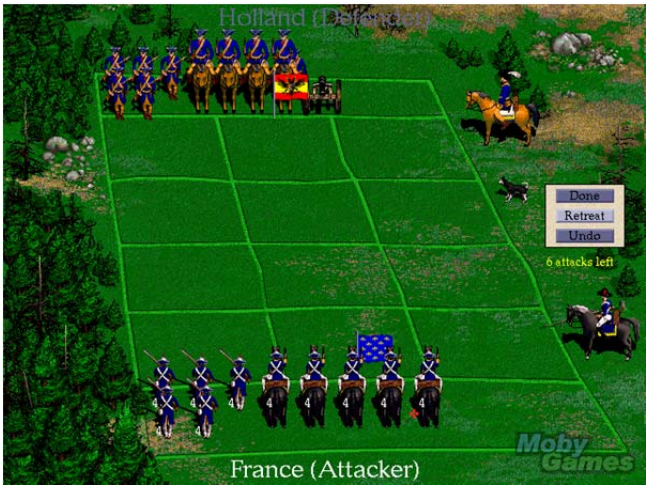
Osredotočanje na en koncept

Vidimo, da igre postanejo vedno bolj kompleksne in je potrebno iz njih, vsaj za fazo prototipa izluščiti bistvo. Tudi med trenutno aktualnimi naslovi je najti kar nekaj takih, ki so iz kakšne stare iger vzeli samo en koncept in se osredotočili samo na njega.

Za primer poglejmo Tower defense, ki je celoten žanr iger, katerih popularnost se je občutno povečala v zadnjih letih. Vendar ideja sama sega že v začetek devetdesetih, konkretno v del Atarijeve igre Rampart. Ostali so s časom rafinirali igralnost in se vedno bolj usmerjali v samo bistvo postavljanja obrambe ter tako ustvarili samostojen žanr, ki so ga ponovno začeli širiti v različne nove smeri.

Na podoben način bi bilo mogoče iz igre Need for Speed: Underground, ki je prinesla vrsto dirke Drag race, vzeti samo ta način in okoli njega osnovati celotno igro. Ali pa recimo iz Civilizacije vzeti samo model vojskovanja in narediti preprosto strategijo, ki recimo precej spominja na Advance Wars,

oziroma če gremo še malce dlje v preteklost, na vojskovanje v Conqeust of the New World, ki bi lahko bil otrok sparjenja bojevanja v North & South s šahom.



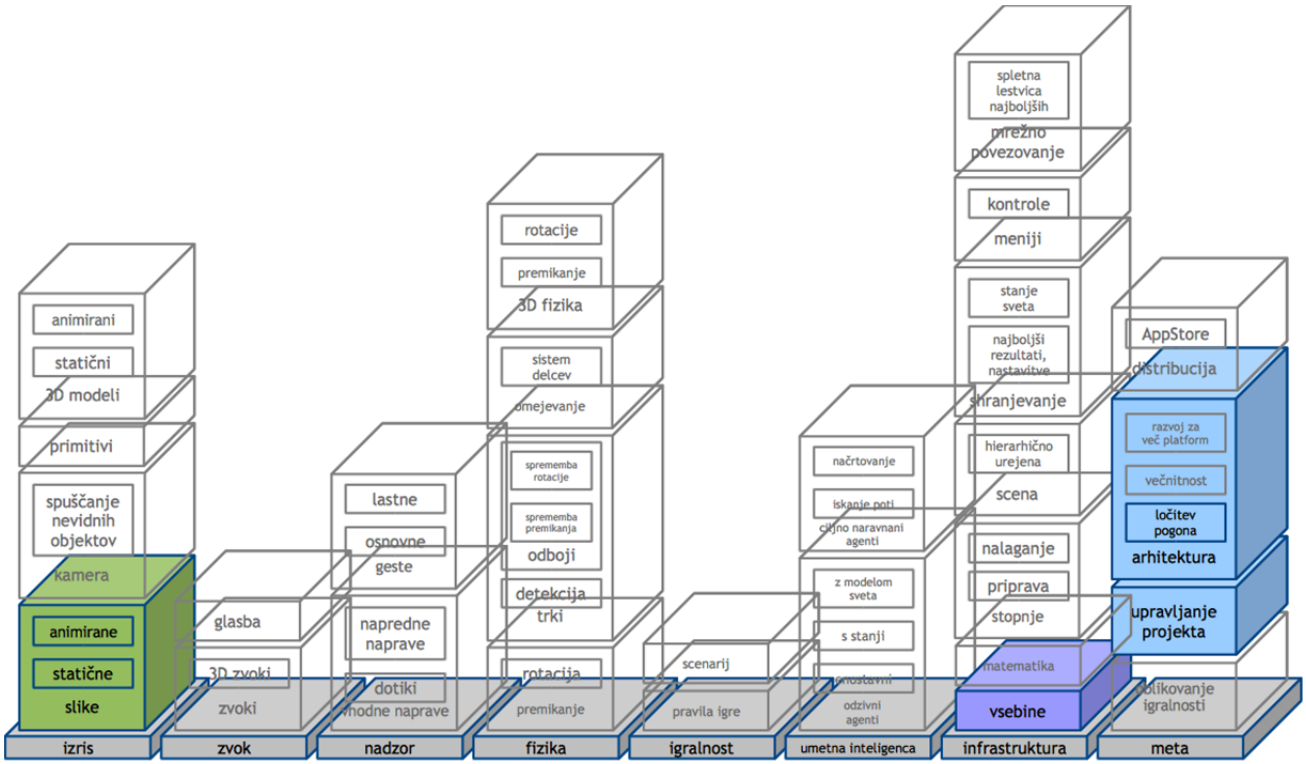
In tako smo že pri drugem pristopu, prenosu ali zgledovanju po namiznih igrah.

Prenos iz prave v navidezno resničnost

Marsikatero namizno igro je mogoče digitalizirati in nam računalnik služi za elektronsko ustreznico igralne plošče, figuric ali kart. Obstaja veliko primerov, recimo prenos iger Enka in Naseljenci otoka Catan na servis Xbox Live Arcade, iPhone in predvsem iPad pa se izkaže za odlično nadomestilo, ko si zaželimo igro šaha, dama ali reversija, pa s sabo nimamo plošče in figur.

Zelo očiten primer so tudi vsi športi, ki so prenešeni v virtualno različico, pa naj bo to igranje badmintona na vrtu za hišo ali simulacija nogometa z menedžerskim delom vred. Konec koncev so dirkanje, padalstvo in streljanje iz prejšnjih primerov vse osnovane na neki resnični dejavnosti. Zato je marsikdaj mogoče dobro idejo dobiti, če pogledamo izven okvirov že narejenega, ter se vprašamo, kako bi kakšen pojav v pravem svetu čim bolje povzeli v kakšni igralni mehaniki.

Sklop 2


animirani
statični
3D modeli
primitivi
spuščanje nevidnih objektov
kamera
animirane
statične
slike
izris
glasba
3D zvoki
zvoki
zvok
lastne
osnovne geste
napredne naprave
dotiki
vhodne naprave
nadzor
rotacije
premikanje
3D fizika
sistem delcev
omejevanje
sprememba rotacije
sprememba premikanja
odboji
detekcija trki
rotacija
premikanje
fizika
scenarij
pravila igre
igralnost
načrtovanje
iskanje poti
cilj no naravnani agenti
z modelom sveta
s stanji
enostavni
odzivni agenti
umetna inteligenca
spletna lestvica najboljših
prežno povezovanje
kontrole
meniji
stanje sveta
najboljši rezultati, nastavitve
shranjevanje
hierarhično urejena
scena
nalaganje
priprava
stopnje
matematika
vsebine
infrastruktura
AppStore distribucija
razvoj za več platform
večnitnost
ločitev pogona
arhitektura
upravljanje projekta
oblikovanje igralnosti
meta

Priprava za delo

Kaj moramo narediti, preden lahko sploh začnemo ustvarjati?

Z izdelanim načrtom za prototip je končno napočil čas, da se vržemo v kodiranje ... skoraj. Najprej se moramo namreč seznaniti z delovnim okoljem, ustvariti projekt in ga povezati z ogrodjem XNI.

Potrebna orodja

Razvoj iOS aplikacij je možen izključno z integriranim razvojnim okoljem (IDE - angl. integrated development environment) Xcode. Je zastonj in prosto dostopno preko Applove strani za razvijalce.

Poleg tega je potrebno namestiti iOS SDK (angl. software development kit, vsa potrebna programska oprema za razvoj), kar je možno po (brezplačni) registraciji.

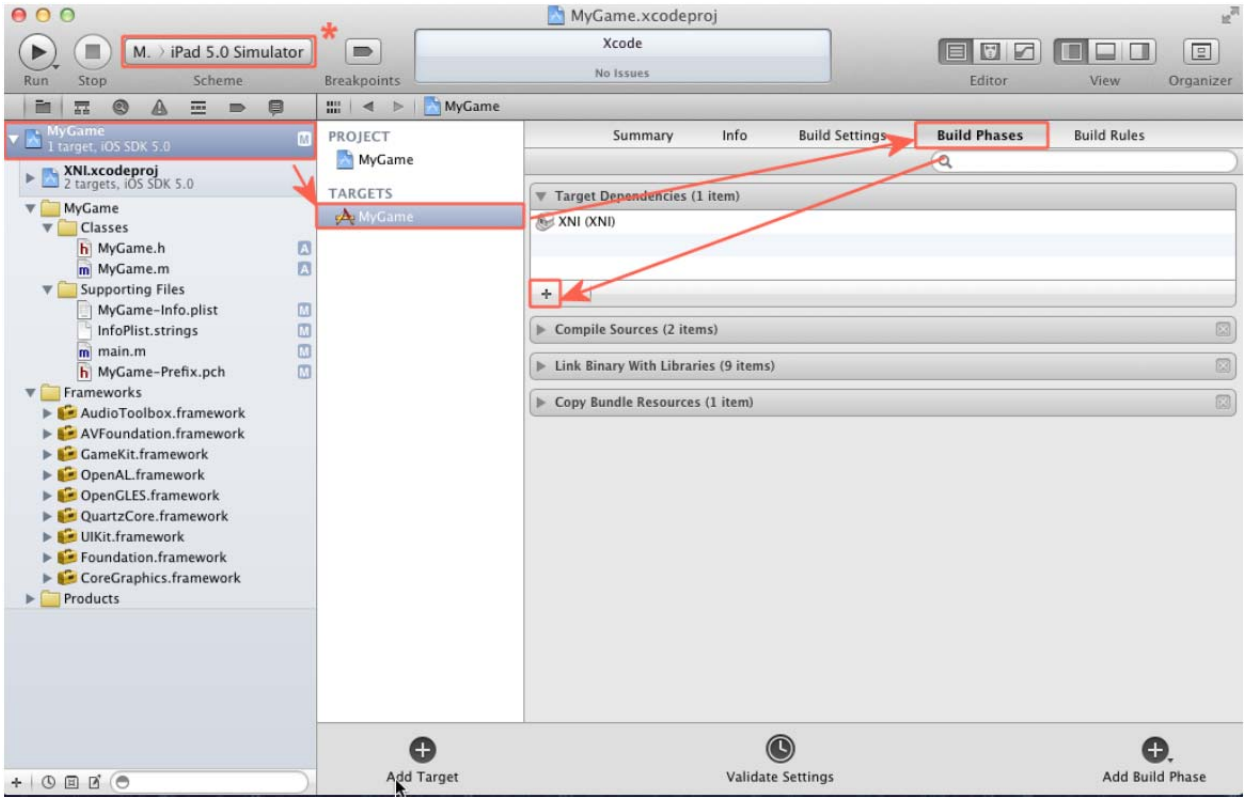
Preden zapremo brskalnik skočimo še po ogrodje XNI. Na desni strani imate pod Featured Downloads na voljo prenos najnovejše verzije v obliki statične knjižnice (libXNI.a skupaj s potrebnimi zaglavnimi datotekami). Navodila za ogled ali prenos izvorne kode preko SVNja najdete na strani Source.

Podrobna navodila za namestitev ogrodja XNI najdete na:
http://code.google.com/p/xni/wiki/StepByStepInstructions

Možnost uporabe izvorne kode XNI

Po uspešnem prenosu kode XNIja iz repozitorija (http://code.google.com/p/xni/source/checkout) imate nekje na disku mapo XNI v kateri je nadalje podmapa `Classes` ter XCode projekt `XNI.xcodeproj`. Tega iz Finderja povlečemo v XCode in spustimo na ime našega projekta (na spodnji sliki npr. `MyGame`) ali pa ga tu z desnim klikom na `Add Existing Files` ročno dodamo. Preverimo, da imamo pri uvažanju izklopljeno možnost `Copy if needed`, pri `Add to Targets` pa mora biti obkljukana naša aplikacija.

Če razpremo na novo dodan projekt `XNI.xcodeproj`, v njem vidimo tudi knjižnico `libXNI.a`. Prevajalniku moramo povedati, da pred prevajanjem naše igre prevede tudi ogrodje XNI. V XCode-u kliknemo na naš projekt in kliknemo na našo igro pod `Targets`. Potem gremo na zavihek `Build Phases`, potem pod `Target Dependencies` kliknemo na plus in dodamo XNI.



Pod `Link Binary With Libraries` je potrebno dodati prevedeno knjižnico XNI (najdemo jo pod razdelkom `Workspace`), poleg tega moramo dodati še vse knjižnice, ki jih XNI potrebuje (seznam potrebnih knjižnic je tukaj).

Za konec je potrebno projektu povedati še, kje lahko najde zaglavne datoteke ogrodja XNI. V XCode-u kliknemo na naš projekt in kliknemo na našo igro pod `Targets`. Potem gremo na zavihek `Build Settings`.
Z iskalnikom si pomagamo najti vrstico `User Header Search Paths`, dvojni klik na vrstico, potem klik na znak + in pod `Path` vpišemo pot do zaglavnih datotek XNIja. Če želimo pot podati relativno na naš projekt, lahko pot projekta najdemo v spremenljivki `"$(SRCROOT)"`.

Preden nadaljujemo, v `Set the active scheme` (na sliki ozančeno z *) preverimo, da imamo izbrano pravilno shemo (lahko prevajamo igro+XNI ali pa samo XNI), in da imamo izbrano platformo Simulator in ne Device.

Zdaj lahko iz menija `Build` izberimo `Build` in knjižnica `libXNI.a` skupaj z vsemi potrebnimi zaglavnimi datotekami bi se morala ustvariti v mapi `XNI/build`.