



Blekinge tekniska högskola
Sektionen för teknokultur, humaniora och samhällsbyggnad
Digitala Spel/ Digital Ljudproduktion
VT08
Kandidatarbete/slutreflektion

VOID

– en spelprototyp för Xbox360

Handledare Mattias Schertell
Examinator Peter Ekdahl
Examinationsdatum, för produktionen och
Slutreflektionen (2008-05-30)

Joakim Lindberg
Fredrik Gullbrandson
Bo Martin Nilsson

jolp05@student.bth.se
frga05@student.bth.se
bmni05@student.bth.se

Förord

I kursen Kandidatarbete på utbildningen Medieteknik med inriktning Digitala Spel och Digital ljudproduktion har vi arbetat med en spelprototyp för Xbox360. Kursen motsvarar 30 högskolepoäng¹ och gjordes under vårterminen 2008. Detta arbete har varit ett samarbete mellan studenter från Digitala Spel och Digitalt Ljud.

Ett stort tack till Jonas Svegländ och Daniel Nilsson som under utbildningen har varit till stor hjälp.

¹ Systemet för högskolepoäng reviderades 1 juli 2007 då en termin tidigare var 20 poäng ändrades det till nuvarande 30 poäng

Abstrakt

Detta examensarbete består utav av två delar, en produktionsdel och denna slutreflektion. Produktionsdelen varade i 15 veckor och gick ut på att utveckla en spelprototyp. Spelprototypen är ämnad för spelkonsolen Xbox360, som är den nya generationens tv-spel. För att kunna utveckla till denna konsol måste man använda sig av XNA, som är ett utvecklingsverktyg ämnat specifikt för Xbox360 och ett måste om man vill kunna utveckla till denna konsol som icke licensierad spelutvecklare. Denna rapport är en sammanfattning av den arbetsprocess vi genomfört

Nyckelord:

Xbox360

XNA

C#

Void

Spel

Digital underhållning

Summary in English

This degree final thesis consists of two parts, one production and this final reflection. The production part lasted over the span of 15 weeks and its goal was to develop a prototype for a game. This prototype was meant for the new generation gaming console, Xbox360. To be able to develop for this console one must use XNA. XNA is a software development kit for non-licensed developers. This document is a summary of the whole work process.

Keywords:

Xbox360

XNA

C#

Void

Games

Digital Entertainment

Innehållsförteckning

Förord	2
Abstrakt	3
Arbetsbeskrivning	6
Projektplan	7
Veckorapporter	8
Början av spelmotorn	8
Motivationsbrist	10
Ny arbetsmiljö	10
Reflektion	11
Projekthantering	11
Projekt- och produktionsprocesser	12
Introduktion	12
Influenser	12
Designdokumentet	15
Spelidé	15
Programvara	16
Inlärningsprocess	16
Början av spelet	17
Fienderna	19
Heads up display (HUD)	20
Shaders	20
Ljudet	21
Bristen på motivation	21
Slutfasen	22
Slutord	25
Bilagor	26
Personlig reflektion över projektarbetet "Void" av Joakim	28
Personlig reflektion över projektarbetet "Void" av Fredrik	30
Personlig reflektion över projektarbetet "Void" av Bo Martin	32
Ordlista	34
Källförteckning	35

Arbetsbeskrivning

Under vårt examensarbete har vi producerat en spelprototyp vid namn Void som går ut på att flyga runt, skjuta ner fiender och samla poäng. Vi hade från början bestämt att vi skulle göra ett spel och hade tidigare undersökt möjligheten att utveckla ett spel för konsol. Eftersom alla tidigare projekt vi har arbetat med under studietiden har utvecklats för PC, så ville vi nu prova på en annan plattform. Valet för oss stod mellan att utveckla för Microsoft Xbox360 eller Nintendo DS. Valet föll på Xbox360, på grund av den faktiskt har ett stort stöd för egenutvecklade spel. Detta stöd finns i form av ett community med stora mängder av information och guider om hur man kommer igång med utvecklingen. För att göra detta så enkelt som möjligt för privatpersoner har Microsoft utvecklat ett mjukvaruverktyg kallat XNA.

Anledningen till att vi inte valde att utveckla till Nintendo DS är att den saknar ett officiellt community och utvecklingsverktyg för privatpersoner. Skulle vi ha stött på problem så hade det funnits ytterst lite hjälp att tillgå i form av exempelkod, guider och programvara. Arbetet bestod av att utveckla en prototyp till ett spel ämnat för Xbox Live Arcade på Xbox360. Xbox Live Arcade är en plattform för spel av småskalig storlek och lägre budget. Dessa spel kräver inget större tidsåtagande av spelaren och bygger på enkla regler.

Efter mycket tankearbete och brainstorming kom vi fram till en spelidé, som vi alla var överens om och lät som ett roligt projekt att arbeta med.

Efter denna process påbörjade vi det viktiga arbetet med att skriva ett designdokument. Detta dokument fungerar som ett styrdokument över spelet med avsikt att tillhandahålla information om allt från grundläggande regler för spelet till dess grafiska stil. Här bestämde vi oss tidigt för att musiken skulle ha ett stort inflytande på spelupplevelsen. Detta skulle ge ett intensivt och upplevelsefullt gameplay. Ett spels gameplay är allt en spelare kan uppfatta, uppleva och utföra i spelet.

För att vi skulle kunna utveckla spelet till Xbox360 så var vi tvungna att lära oss ett nytt programmeringsspråk, nämligen C#, då detta är det enda programmeringsspråk som är kompatibelt med XNA. Vi hade redan goda kunskaper i C++, så det hjälpte oss lite när vi skulle börja lära oss C#. Den största skillnaden mellan de båda språken är deras sätt att hantera minnet. Eftersom C++ tillåter oss att själva styra när vi vill allokeras och fria minnet fungerar C# på ett annorlunda sätt. På grund av att C# tillåter oss att allokeras minnet, tillåter det inte oss att själva styra deallokeringen. Vi kan säga till datorn att fria minnet, men datorn gör det först när den anser att minnet inte används mer. Denna process är effektiv på så sätt att man inte får minnesläckor, men det har sitt pris eftersom detta påverkar prestandan negativt.

Projektplan

Mål

Examensprojektet eller Kandidatarbetet är ett arbete i medieteknik 30 poäng som alla studenter ska göra för att kunna ta sin Kandidatexamen. Arbetet är menat att vara kunskapsutvecklande inom ämnet medieteknik. Det ska vara en gestaltande produktion inom ämnet och tillämpa kunskaper och färdigheter i projekt- och produktionsprocesser. Det ingår även en slutreflektion som ska vara förankrad i relevanta referenser.

Metod

Varje projekt ska tilldelas en fast handledare som under projektets gång ska hålla regelbundna delexaminationer av idéarbete och resultat. Dessa delexaminationer ska utgå från gruppens veckorapporter och projektplan. Kursen kommer även att kompletteras med relevanta föreläsningar och seminarier.

Examinationen för Kandidatarbetet är uppdelat i tre delar:

Delexaminationer under projektets gång gällande kunskapsutvecklingen.
En offentlig examination med opponenter gällande tillkomsthistoria.
Examination av slutreflektionen.

Vår projektplan finns att tillgå som bilaga i slutet av denna reflektion.

Veckorapporter

Början av arbetet

Vi började med att försöka få en gemensam vision av det spel vi ville göra. Vi testade lite olika Xbox Live Arcade titlar i detta syfte, dessa var Ikaruga (2001, Utvecklare Treasure Co. Ltd.), Geometry Wars (2005, Utvecklare Bizzare Creations), Mutant Storm Reloaded (2005, Pom Pom Games). Tidigare har vi testat Boom Boom Rocket(2007, Bizzare Creations) och Rez (2001, Sega), vilka var anledningen till att vi från början ville göra ett spel där musiken har ett stort inflytande på spelupplevelsen.

Eftersom vi var i ett tidigt stadium av projektet hade vi inte delat in arbetsområdena ännu. Fredrik Gullbrandson och Joakim Lindberg lärde sig mer om C# och XNA medans Bo Martin Nilsson lärde sig hur trackerbaserad musik fungerade, ett sätt att programmera musik, gick till. Programmeringsspråket och sättet att skapa musik var nytt och spännande och det kändes som att vi hade kontroll över det mesta, vilket resulterade i en relativt hög ambitionsnivå.

Efterhand som projektet fortskred upptäckte vi nya idéer för att uppnå vår vision av hur spelet skulle se ut. Vi letade hela tiden efter nya idéer i guider och exempelkod (Microsoft, <http://creators.xna.com/education/catalog/>) som fanns att tillgå genom internet. Fredrik Gullbrandsson och Joakim Lindberg hade också fått större förståelse för det nya programmeringsspråket och Bo Martin Nilsson hade kommit på nya idéer och skisser för spelmusik och ljudeffekter.

Sjukdomar satte käppar i hjulen för oss och vi förlorade en del arbetskraft och tid. Detta medgav en hel del extra arbete och vi försökte arbeta in så mycket som möjligt för att vinna tillbaka den tid vi förlorat.

Efter hand som vi lärde oss mer om XNA, och påbörjade uppbyggnaden av spelmotorarkitekturen, förstod vi hur mycket mer vi var tvungna att lära oss. XNA skiljde sig en del mot vad vi arbetat med innan och det var dessa skillnader som vi behövde vänja oss vid. Detta gjorde att vi fick lägga ytterligare tid på efterforskning, men vi hade dock skapat en viss arbetsfördelning för att effektivisera inlärningsprocessen. Då vi enbart forskade i hur C# och XNA fungerade började dock motivationen sjunka. Vi ville börja utvecklingen av spelet men hade inte tillräckliga kunskaper. Detta gjorde oss mycket frustrerade. Bosse visade dock koncept på en låt som skulle kunna vara i spelet, vilket gav tillräckligt med inspiration för att ge energi och nya krafter.

Början av spelmotorn

När utvecklingen av spelmotorn hade börjat på allvar stötte vi på svårlösta problem. Rotationerna av 3D-objekten fungerade inte korrekt i XNA. När ett 3d-objekt lästes in stämde inte dess rotation, vilket gjorde att vi fick rotera den rätt i spelmotorn. 3d-objekten var spegelvända och roterade 90 grader runt X-axeln. Detta skapade ytterligare ett problem då rotationen runt Z-axeln helt ologiskt slogs ihop med Y-axeln, modellen roterade således på

samma sätt oavsett vilken av dessa axlar man valde att roterade runt. Vi löste detta efter många dagars resultatlösa efterforskning, genom att från 3D-modelleringsprogrammet exportera modellerna felroterade, alltså spegelvända och 90 grader runt X-axeln.

Samtidigt som vi hade dessa problem arbetade vi parallellt med renderingssystemet. Detta gav också problem då vi försökte använda våra egna shaders, istället för de inbyggda i XNA. Vi sökte i internet forum efter lösningar och gjorde många små förändringar innan vi beslutade oss för att skriva om hela renderingssystemet, vilket visade sig lösa problemet utan förklaring.

När rotationsproblemet var "löst" lade vi in en tidig version av ljudmotorn för att se ifall den fungerade med resten av koden. En förfinad arbetsrutin för musikskapandet togs fram för att snabbare kunna skapa färdig musik för spelet. Detta genom att en uppsättning passande instrument togs fram för att hålla en viss enhetlighet i spelmusiken, d.v.s. att man håller sig till en viss uppsättning instrument i all musik.

Vi började fundera på hur kollisioner mellan objekt i spelvärlden skulle beräknas och använde oss av internet för att leta upp färdiga kollisionsbibliotek, (*Ship Game: 3D Collision with the BoxCollider Library, 2007, Microsoft*) och (*Ageia PhysX, http://www.nvidia.com/object/nvidia_physx.html*), men fann att dessa antingen var för komplexa eller saknade den funktionalitet vi behövde för vårt spel. Detta resulterade i att vi programmerade ett eget kollisionssystem som enbart innehöll de delar som vi behövde, nämligen att enkelt kunna få information om när objekt kolliderar med varandra och vilka som kolliderade.

Extensible Markup Language (XML) är en språkstandard för att strukturera data och vi använde denna för att läsa in information till spelet. T.ex. vart spelaren startar på skärmen, hur mycket liv man börjar med o.s.v. Vi visste även att C# hade inbyggt stöd för hantering av XML filer. Detta skulle komma att underlätta vårt arbete vid scriptandet av banor och fiender. Att scripta i detta faller innebär att editera XML filerna. Vi upptäckte även att man kunde använda sig av XNA för att snabba upp inläsningen av dessa filer men man var då tvungen att skriva en egen pre-processor som hade hand om detta.

En pre-processor är ett program som bearbetar inläsningsdata för att sedan kunna göra om det till data som ett annat program kan förstå. Detta används för att snabba upp inläsningsprocessen. Vi beslutade att vi inte hade tid att sätta oss in i hur det fungerade att skriva en pre-processor och använde oss istället av C# inbyggda funktioner som var mindre effektiva.

Under tiden som vi jobbade med XML-inläsningen gjordes även menysystemet för spelet. En del omstrukturering av spelmotorn gjordes även under tiden. Detta innebar att spelmotorn nu gjordes om så att den fick ett antal olika tillstånd. När spelet precis startats laddades t.ex. menytilståndet. Gjorde man sedan ett menyval så laddades det valda menyvalets tillstånd. Detta kallas att spelmotorn är *State driven*.

När vi börjat få ordning på kollisionssystemet för spelare och fiender, samt gjort ett användargränssnitt började spelet kännas mer komplett, vilket var en stor inspiration vid musikskapandet. Att skapa musik med rätt känsla, utan att ha något att relatera till, hade nämligen visat sig vara mycket svårt. Sköldar och fiender hade också tagit form och det kändes som vi verkligen hade något på gång.

Vi skapade en weaponmanager som har hand om alla vapnen och implementerade en funktion som gjorde att man kunde skjuta. Det gjordes även klass som hade hand om inläsningen av banor. Den klassen hade i sin tur hand om olika spawn points som skapade fienderna och placerade ut på spelplanen. Nu var de huvudsakliga funktionerna för gameplay klara. Man kan flyga runt på banan, skydda sig med sin sköld och skjuta.

Motivationsbrist

Efter de problem med vi haft med bland annat rotationer och andra saker som inte borde vara fel, men var det, så hade vår motivation har fått sig en rejäl törn. Det blev inte mycket programmerat under de veckor motivationsbristen var som starkast. Vi spelade in ljud i högskolans studio för att använda som ljudeffekter i vårt spel. Detta livade upp stämningen när vi fick komma ifrån kodandet ett tag.

Trots att vi spelade in ljud och fick lite annorlunda arbetsuppgifter ett tag, så blev motivationsbristen ändå så pass allvarlig att vi bestämde oss för att ta en vecka ledigt för att kunna koppla av och göra lite roligare saker än att försöka rätta till buggar och andra konstiga fel.

När vi väl kom tillbaka så hade vi inte hittat den gnista som vi hoppats skulle ge oss motivation för att kunna fortsätta arbetet. Vi satt nästan i en vecka i högskolans lokaler utan att göra några nämnvärda framsteg, efter detta beslöt vi oss för att inte sitta i högskolan och arbeta. Vi behövde byta arbetsmiljö.

Ny arbetsmiljö

Vi flyttade vår arbetsplats från högskolan till Joakim Lindbergs lägenhet. Detta miljöombyte gav oss den kraft, som behövdes för att åter igen ta tag i arbetet. Alla de småsaker som vi haft problem med löste vi ganska kvickt och vi fick mycket gjort de sista veckorna. Till exempel löste vi så att man fick poäng då man sköt ner fienden. Ett partikelsystem skrevs för att kunna ge respons då man dödade en fiende samt att man fick se sin poäng visas i text vid fienden.

När vi hade fått ett fullt spelbart spel, ansåg vi att den grafiska stilen inte passade, och efter diskussion bestämde vi oss för att göra den mer abstrakt. Vi gjorde till exempel om modellen till spelaren och hela användargränssnittet samt satte en aning annorlunda design på den omgivning spelet utspelade sig i. Vi lade också in ljudeffekter och musik, vilket visade sig skapa några oväntade problem. Bland annat så fungerade inte riktigt verktygen för kompilering av ljudbankerna korrekt, samt att det kunde blir distorsion då det hände mycket på skärmen. Dessa problem är ännu inte lösta (maj 2008).

Reflektion

Projekthantering

Vår grupp har tidigare arbetat tillsammans, då vi våren 2007 hade en produktionskurs vars mål var att fungera som en skarp produktion. Vi arbetade då i ett team på arton personer där alla hade sin bestämda roll. Rollerna bestod av Producent, Lead Design, Lead Art, Lead Sound, Lead Tech och under varje Lead arbetade ett antal personer med uppgifter som delegerats. Det hölls regelbundna möte i början av varje vecka där Producenten och Leads bestämde vilka mål som skulle gälla för veckan.

Något som vi lärde oss i denna produktionskurs var att hur bra man än planerar i förväg så kommer det vara ytterst svårt att följa, vilket vi även kom att uppleva i examensarbetet.

För att i spelet kunna efterlikna någonting som spelaren kan relatera till behövs någon typ av grafiska objekt. Numera kodar man inte grafiken som man gjorde förr, man använder sig av externa verktyg och program för att skapa digitala illustrationer som man sedan använder sig av i sin spelkod. Komplexiteten i dessa program kräver dessvärre en utbildningskurs i sig, vilket saknas i de utbildningsprogram vi läser.

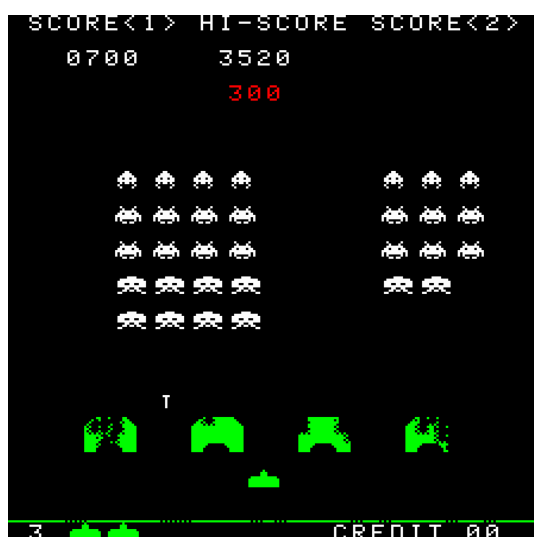
Detta var också en lärdom vi fick av produktionskursen. Bristen av grafiker på högskolan, eller snarare den totala avsaknaden av dem är stor. Detta gjorde att vi valde att göra ett spel utan ett komplext grafiskt innehåll och att vi höll oss till relativt enkla geometriska former för fiender och spelare.

Projekt- och produktionsprocesser

Introduktion

Det första vi gjorde var att bolla idéer med varandra om vad för typ av spel vi skulle göra. Vi delade alla samma vision om att musiken skulle vara en central del av spelupplevelsen. När vi väl klargjort detta så försökte vi komma på en passande spelgenre. Valet föll på den klassiska arkadspelsgenren, shoot 'em up.

Ett av de tidigaste exemplen inom den genren är spelet Space Invaders (1978, Taito Corporation), där man såg sitt spelarskepp samt fienderna ovanifrån. Målet med Space Invaders var att skjuta ner alla fiender som man såg på skärmen, samt akta sig för fiendens eldgivning.



En bild från arkadversionen av Space Invaders, källa: <http://www.kudla.org/raindog/si.html>

Vårt spel bygger på en liknande princip där huvudmålet är att skjuta ner fiender och samla poäng.

Influenser

Innan examensarbetet överhuvudtaget började visste vi att vi ville göra ett spel där musiken hade en betydande roll för spelupplevelsen. Det var främst ett spel som hade stort inflytande på oss, nämligen Boom Boom Rocket (2007, Bizzare Creations).

Boom Boom Rocket är ett spel där man ska smälla av fyrverkerier i takt med musiken. Det är ett enkelt koncept men trots det roligt och beroendeframkallande. Att spela detta spel utan musik är som att spela på en gitarr utan strängar. Vi ville att musiken i vårt spel skulle ha lika stor roll.

Ett annat spel som använder sig av musiken är det kultförklarade Rez (2001, Sega). Man flyger runt i ett virtuellt datorsystem som representeras av enkla geometriska former, där ens

mål är att markera objekt som ska tas bort. Dessa dör sedan och skapar rytmer som spelas i takt med musiken.

Boom Boom Rocket



Källa: www.gamespot.com

Rez

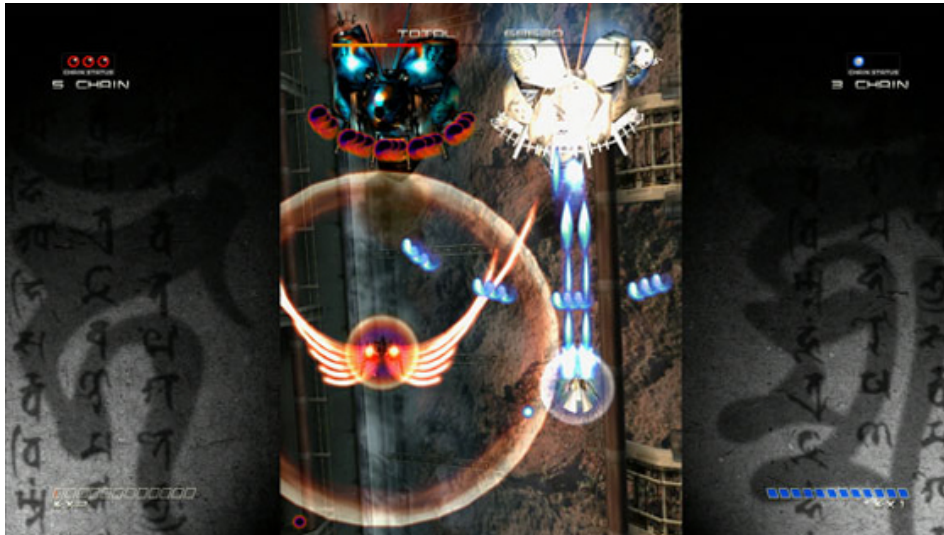


Källa: www.1up.com

Vi gjorde efterforskningar om hur spelscenen ser ut i dag. Vi spelade en mängd spel i samma genre som fanns att ladda hem från Xbox Live Arcade. Däribland fann vi några spel som var mer intressanta än andra. De spel som vi tyckte mest om var Geometry Wars (2005, Bizzare Creations), Ikaruga(2001, Treasure Co. Ltd.) och Mutant storm Reloaded (2005, Pom Pom Games).

Ikaruga utvecklat av Treasure Co. Ltd

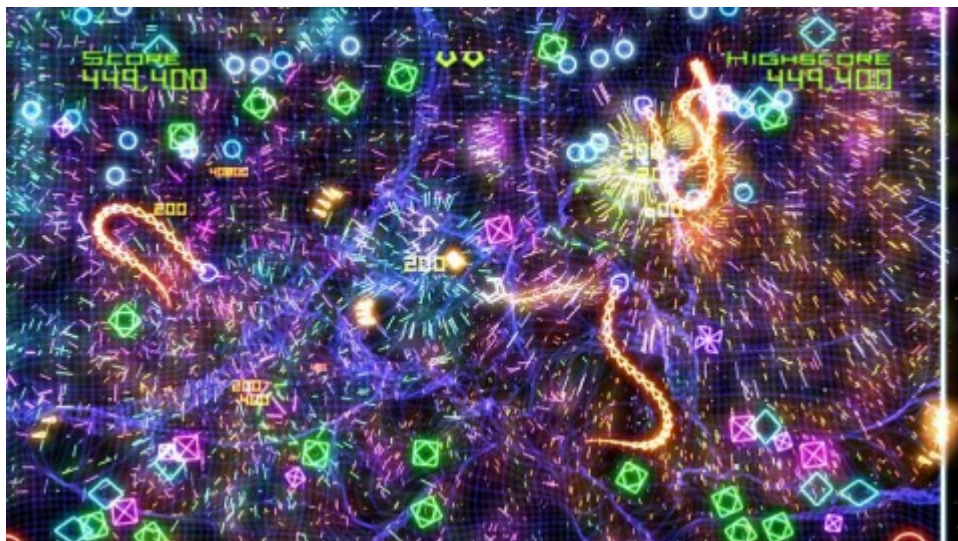
Detta är ett spel som vi fastnade ganska tidigt för. Det har ett intressant spelmoment där skeppets sköld har en betydande roll i spelets gameplay. Dess funktion är att absorbera fiendens skott av samma färg och det finns två olika färger, svart och vitt. Samtidigt som man måste vara snabb på att skifta mellan dessa, måste man även skjuta ner fienden.



källa: <http://www.dorkclub.com/ikaruga-now-available-on-xbox-live-arcade>

Geometry Wars utvecklat av Bizarre Creations

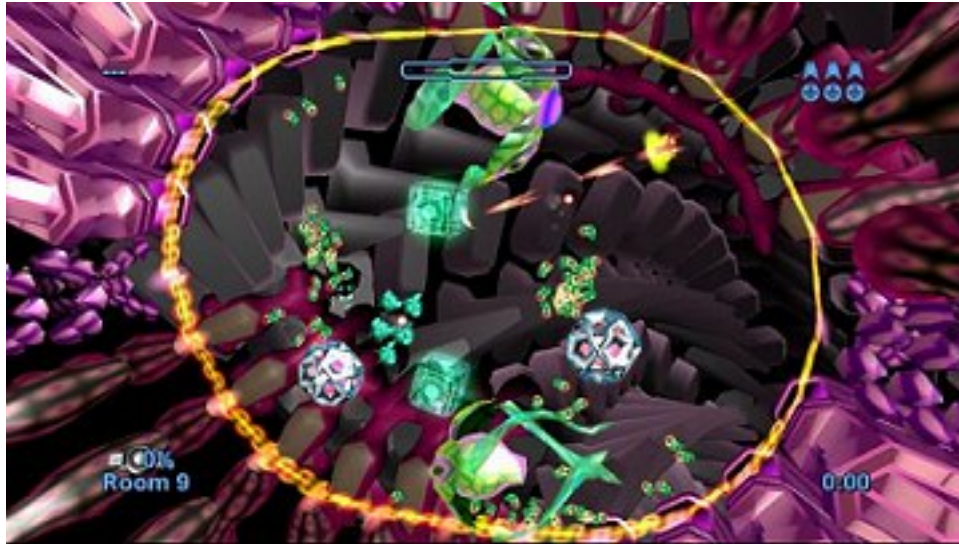
De delar som inspirerade oss mest från Geometry Wars var dess färgglada partikelsystem, spelarskeppets kontroll och eldgivning. Man kan nämligen skjuta 360 grader omkring sig, till skillnad från Ikaruga där man bara kan skjuta rakt fram, vilket ger ytterligare en dimension. Spelaren har bara en spelplan att röra sig på och spelet slutar först när spelarens liv har nått noll.



http://games.gearlive.com/blogimages/geo_wars_thumb.jpg

Mutant Storm Reloaded utvecklat av Pom Pom

I likhet med *Geometry Wars* så har även detta spel 360 graders eldgivningsfrihet, men ens rörelseyta är betydligt mer begränsad. Det finns multipla banor och när man klarat en går man vidare till nästa.



Källa: <http://www.gametunnel.com/articles.php?id=273>

Design dokumentet

När dessa ramar var bestämda så började vi skissa på ett designdokument där vi hade all information samlad . På så sätt hade vi någonting att falla tillbaka på om något blev oklart senare under arbetsprocessen.

Vårt designdokument för detta projekt innehöll först en presentation av vår spelidé, spelförlopp, spelmekanik, användargränssnitt och inspirationskällor. Det innehöll även beskrivning av föremål som har positivt eller negativ effekt på spelaren. Exempel på en positiv effekt är att öka spelarens rörelsehastighet, så att denne lättare kan undvika fiender. En negativ skulle då vara en motsatt effekt och göra spelarens skepp långsammare.

Spelidé

Som vi innan nämnt så ville vi att musiken skulle ha en stor roll i spelet och att vårt spel skulle vara en form av shoot'em up. Vi gav detta spel namnet Void. Namnet betyder ungefär tomrum.

Spelet går ut på att spelaren kontrollerar ett rymdskepp med fyra olika sorters sköldar. Sköldarna har färgerna röd, grön, gul och blå. Dessa sköldar aktiveras när spelaren trycker på en knapp med motsvarande färg på handkontrollen. Dock kan skölden bara användas en kort stund och måste därefter låtas att ladda upp sig själv, vilket görs genom att inte använda skölden.

Spelaren befinner sig i en värld som ständigt skiftar färg. I denna värld dyker det upp fiender som på något sätt försöker flyga in i spelaren. Dessa fiender kan ha fyra olika färger och om de lyckas köra in i spelaren så förlorar spelaren energi, som är spelarens hälsa. När energimätaren står på noll så dör spelaren. Skulle spelaren lyckas att skydda sig med sin sköld med samma färg som fienden så absorberas fiendens energi och denna dör. Energin som absorberas från fienden läggs på till spelarens energi.

När spelaren inte använder sig av sin sköld så kan han/hon även skjuta. Om man skjuter ner en fiende får man poäng. Poängen man får beror på hur mycket energi spelaren har fått genom att tidigare absorbera fienden. Spelet går alltså ut på att samla så hög poäng som möjligt och detta görs genom att finna en så bra balansgång som möjligt mellan att absorbera och skjuta ner fienden.

Vi tänkte ha flera banor i spelet där varje bana avslutades med en boss. En boss är en unik fiende med speciella attackmönster och kräver taktiskt tänkande för att bemästra. En bossfight ska vara ett avslutande moment av en bana, med syfte vid seger att ge en känsla av framgång och triumf.

Programvara

Nästa steg för att arbetet skulle fortsätta var att införskaffa de nödvändiga program och licenser. Vi skaffade oss ett medlemskap i XNA Creators Club vilket gav oss tillgång till exempelprogram och färdiga spel vi kunde ta lärdom av.

Andra program som behövdes var Visual Studio C# Express(2007, Microsoft), som är den utvecklingsmiljö vi arbetade i. Vi behövde även XNA som är det application programming interface (API) man använder till Visual Studio för att utveckla mot Xbox360. Ett API fungerar som en mellanhand mellan hårdvara och utvecklingsverktyg, vilket underlättar för programmeraren.

Inlärningsprocess

Vi kunde inte börja arbeta med spelet riktigt än, då vi varken hade kunskap om C# eller XNA. Så vi var tvungna att först ha en inlärningsperiod om hur C# fungerade och fick läsa grundläggande böcker och guider för att tillgodose oss den kunskap vi behövde. Den bok vi använde för C# var Harris, A.(2002). *C# programming for absolute beginners*. Boston:Course PTR. Vi fann den via BTHs elektroniska biblioteket, *ebrary*. Efter att ha läst ca 200 sidor och gjort diverse övningar kände vi oss självsäkra nog att ge oss in på XNA istället. Detta var ytterligare ett steg som var tvunget att tas innan vi kunde påbörja arbetet med vårt eget spel. Vi gick igenom en del av de guider som kan finnas på creators.xna.com, däribland en guide för hur man skapar ett spel på en timme (<http://msdn.microsoft.com/en-us/library/bb975644.aspx>), där författarna gick igenom de grundläggande nödvändiga kunskaperna.

Under tiden funderade ljudansvarig på en enkel metod för att synkronisera musik med

spelförlopp, samt lämpligt filformat för musiken åt detta. Tester gjordes med trackermjukvaran Renoise(2000, Eduard Mueller och Zvonko Tesic), eftersom det är lättare att läsa av taktslag ur en trackerlåt, samt att de är mindre utrymmeskrävande. Anledningen till detta är att en trackerfil endast innehåller notdata², effektdata samt en sampling³ av de olika instrumenten som ingår i låten. Detta gör filerna små och notdatan gör det enklare att synkronisera uppspelning med händelser i spelet. Dessvärre hade inte Xact – ljudmotorn för XNA – stöd för trackermusik och efter att ha upptäckt detta funderade vi på alternativa lösningar. Det visade sig att Xact enbart hade stöd för vanliga PCM-ljudfiler – Pulse-code Modulation, som innehåller enbart ljuddata – så en mer avancerad implementering av taktigenkänning var nödvändig för att vår ursprungliga spelidé skulle fungera. Detta skulle dock ta för lång tid att göra så vi beslutade oss för att överge våra ursprungliga planer vad gällde musikens påverkan av spelförloppet.

Vid denna tidpunkt i projektet kände vi oss tillräckligt redo för att faktiskt påbörja arbetet på spelet. Dock återstod en sak att göra, vi var tvungna att skriva ett techdokument. Techdokumentet ska innehålla vilka delar som programmeringsmässigt måste finnas med och kan tänkas behövas. Det innehåller spelmotorns arkitektur med de olika klasser samt metoder och attribut klasserna ska innehålla.

Början av spelet

När all förundersökning och efterforskning var klar kunde vi påbörja utvecklandet av spelet. Ljudansvarige satte sig ner och började skissa på musik samt ljudeffekter som han ansåg skulle passa till spelet, med riktlinjer utifrån designdokument samt inspiration från liknande spel. Musiken testades sedan i ett spel vid namn Audiosurf.



AudioSurf utvecklat av Invisible Handlebar, Källa: <http://www.audio-surf.com/>

Audiosurf är ett futuristiskt racingspel där vägen man färdas på byggs upp av musiken.

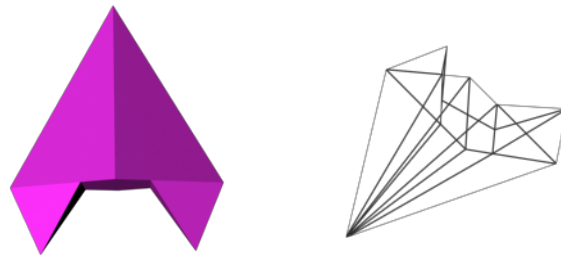
² Notdata innehåller vilken not som ska spelas upp

³ En sampling är en inspelning av ett instrument.

Vilken metod som används för att läsa av musiken i Audiosurf är för oss okänt, men ett sätt att leta upp t.ex. basdunk i en låt är att leta efter transienter som följs av en vågform med låg frekvens. En transient är en plötslig ökning av ljudstyrkan i ett ljud. Det kan t.ex. vara en ett anslag mot ett truminstrument eller bara en enkel handklappning.

I spelet Audiosurf styrs den hastighet man färdas med av takten. Olikfärgade kvadrater, vilka man ska samla på, byggs sedan upp utefter händelser i musikarrangemanget, vilket ger en känsla av att man färdas i musiken. Audiosurf var på så sätt ett effektivt verktyg för att testa ifall musiken hade alla ingredienser som skulle behövas för att passa in i ett spel. Efterhand som spelet tog form övergavs dock Audiosurf och ersattes av en tidig prototyp av vårt egna spel.

Vi hade tidigt bestämt oss för att ha enkla geometriska former för spelare och fiender, så vi började skissa upp en prototyp av spelarskeppet. Det skulle vara något som visade att den som spelar styr ett rymdskepp. Vi hittade en form som vi var nöjda med, skeppet var som en triangel så att man inte kunde ta miste om från vilket håll spelaren styrde.



Ovan ser vi det spelarskepps koncept som vi kom fram till, vänster en solid form medan till höger ser vi skeppet som en trådmodell (wireframe).

När vi hade fått in spelarmodellen i vår motor så hade vi problem med att få skeppet rätt roterat, antagligen är det något i XNA som försvårar rotationer med tanke på att vi inte har haft problem med rotationer tidigare. I sökandet efter en lösning på detta problem gick det åt mycket tid.

För att överhuvudtaget ha ett gameplay så måste man ha en fungerande kollision mellan objekt. Vi tittade på olika kollisionsbibliotek som fanns att tillgå på internet, (*Ship Game: 3D Collision with the BoxCollider Library, 2007, Microsoft*) och (*PhysX, Ageia*). Tidigare hade vi arbetat med Ageia PhysX som är ett av de ledande API⁴ för kollision på marknaden. Hade vi använt detta så skulle vi bara ha använt oss av en bråkdel av dess kapacitet. Då man kan göra ytterst avancerade kollisionsberäkningar med detta, vilket skulle ha varit överflödigt för vårt projekt. Därför sökte vi efter alternativa API, bland annat så följde det med ett bibliotek till ett av de spel som vi fick genom vårt Creators Club medlemskap. Detta bibliotek saknade dock viss funktionalitet som vi behövde använda oss av i vårt projekt.

Slutligen bestämde vi oss för att ta saken i egna händer och programmera ett

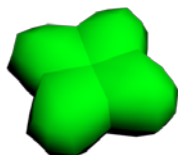
⁴ Application Programming Interface, Ett API fungerar som en mellanhand mellan hårdvara och utvecklingsverktyg

kollisionsbibliotek själva. Detta gick relativt smidigt och vi började känna att vi hade ett spel på gång när man kunde åka in i objekt och få en respons tillbaka. I detta fall skrevs det ut en text på skärmen när en kollision mellan två objekt inträffade.

Nu när kollisionen var färdig så ville vi ha fiender att kollidera med, vi satte oss ännu en gång vid ritblocket och skissade upp våra första fiender. Återigen använde vi oss av enkla geometriska former. Vi ville att varje fiende i spelet skulle ha ett unikt utseende så att de lätt skulle kunna skiljas åt, med tanke på att en del fiender kunde ha samma färg. Spelaren skulle lätt kunna identifiera vilken typ av fiende det är genom dess form eftersom varje typ av fiende hade ett eget beteendemönster som krävde olika sorters taktik att bemästra.

Fienderna

Pinch



Pinch har ett slumpmässigt rörelsemönster och är den lättaste fienden av dem alla. Den rör sig långsamt och dör av ett enstaka eller ett fåtal skott beroende på vapen. Den gör heller inte särskilt mycket skada om den skulle köra in i spelaren.

Smudge

Smudge rör sig mot spelarens position och ju närmare Smudge kommer spelaren desto högre hastighet får den. Detta gör Smudge till en ganska svår fiende som dessutom tål en hel del skott.



Blur

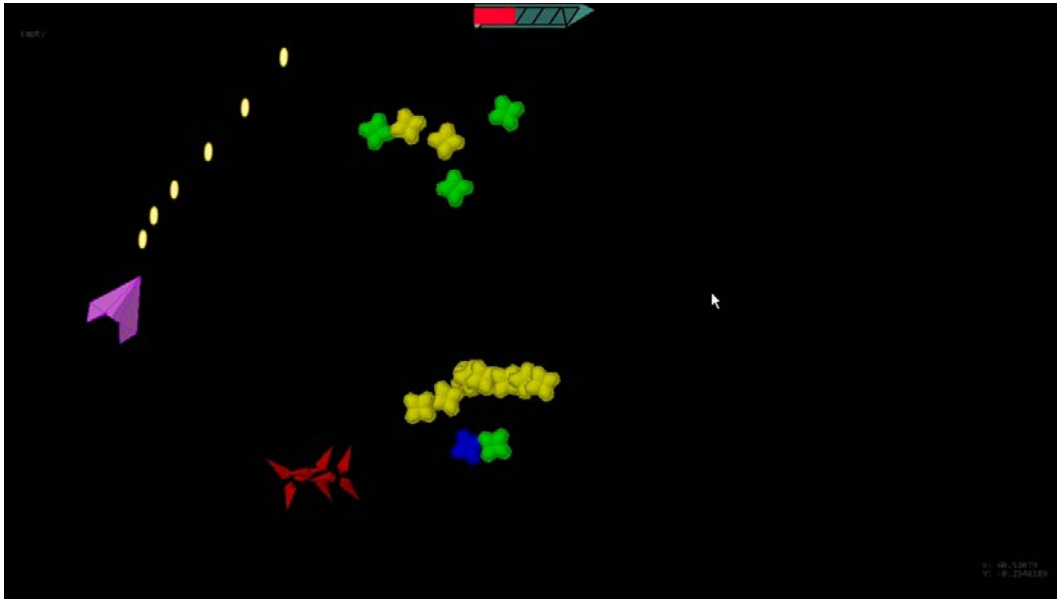


Blur är fienden som gör att spelaren inte kan stå still speciellt länge, eftersom den tar sikte på spelaren och åker med hög hastighet så att man blir tvungen att flytta på sig. Blur tål en hel del skott så, vilket gör den till en av spelets svåraste fiender.

Bump

Denna fiende är designad som en miniboss, en sällsynt svår specialfiende som kan dyka upp innan slutbossen. När vi programmerade denna fiende hade vi återigen problem med rotationer, vilket var något som följde med oss under hela projektet. Detta bidrog till att vi inte har med denna fiende i spelet.





Så här såg vårt spel ut i detta tidiga stadium.

Heads up display (HUD)

För att kunna se hur mycket hälsa och poäng spelaren har och få någon sorts respons på vad som händer i spelet, så behövs en HUD. Vi gjorde en temporär HUD ganska tidigt för att lätt kunna felsöka den information vi ville övervaka genom att rita ut informationen på skärmen via HUDen. Vi observerade i Rez HD att spelet gav feedback utifrån spelarens handlingar, i form av text som ritades ut i bakgrunden. Vi tyckte att det var en intressant idé och provade detta och tyckte att det passade in så vi beslöt oss för att ha kvar det⁵.

Shaders

För att få till snygga effekter på spelaren och fienderna behövde vi skriva shaders. Vi hade redan kunskaper i High Level Shader Language(HLSL) som är ett programmeringsspråk som används för att koda shaders. En shader är en uppsättning instruktioner som skickas till grafikkortet för att kunna rita ut (rendera) all grafik på skärmen. Detta kan även användas för att lägga olika typer av effekter på de objekt som ska renderas. Man kan till exempel lägga tv-brus eller lägga på en effekt som gör så att objektet ser ut att vara ur fokus. Det är bara fantasin och matematikkunskaperna som sätter gränserna.

Det fanns redan inbyggda shaders i XNA, men dessa tillät oss inte att använda de effekter som vi önskade. Vi ville ha upplysta konturer på våra objekt, men en sådan shader fanns inte att tillgå. Av denna anledning var vi tvungna att skriva våra egna.

Återigen stötte vi på problem, då inläsningen av dessa shaders skiljde sig från andra API vi tidigare arbetat med. Vi fick dem inte att laddas korrekt och då renderades inte objekten ut på skärmen. Efter mycket om och men beslöt vi oss för att skriva om renderingssystemet som hanterade våra shaders. Detta löste problemet och vi vet inte riktigt vad det var som gjorde att renderingen fungerade, för i grund och botten var det precis samma sak som tidigare.

⁵ Resultatet av detta syns i bilden längst ner på sidan 22



Till vänster ser vi vår fiende utan "ur fokus" shader och till höger med. Detta är bara ett exempel och används aldrig i vårt spel.

Ljudet

En erfarenhet som vi hade med oss från tidigare spelproduktioner var att filma spelprototypen utan ljud, vilket sedan skulle ljudläggas likt man ljudlägger en film. Detta gjorde att vi tidigt kunde se ifall musik och ljudeffekter fungerade ihop med spelet. Vi kunde också tidigt ta beslut om hur implementationen av ljudmotorn skulle ske utifrån de effekter vi använt oss av vid ljudläggningen av spelscenariot.

Vår idé var att musiken i spelet skulle påverka spelupplevelsen på så sätt att allt visuellt påverkades av takt och ton. Fiender skulle pulsera i takt och dynamik samt tempo skulle påverka antalet och typen av fiender, som dyker upp för spelaren. Spelaren skulle även belönas av att utföra saker i takt med musiken genom att de skulle få extra poäng samt respons genom musiken i form av högre intensitet och tempo.

Vår ursprungliga idé lades på hyllan då vi upptäckte begränsningarna i den ljudmotor som erbjöds av XNA. Begränsningarna utgjordes av brister i ljudformatskompatibilitet. Att göra dynamiskt förändrande musik hade krävts någon form av filformat, som lagrar notdata och enskilda samplningar av instrument. Då hade vi kunnat göra transponeringar samt flyttat om takter och stycken i realtid för att bemöta spelarens handlande i spelet. Begränsningar låg också i det utrymme som erbjöds för publicering av spelet på Xbox Live Arcade. Med ett filformat, som lagrar notdata och enskilda samplningar av instrument, hade vi också kunnat få ner utrymmeskraven avsevärt. Det fanns alltså inte stöd för alla de idéer vi planerat för, utan vi hade varit tvungna att utveckla en egen ljudmotor. Detta fanns det inte tid för.

Bristen på motivation

För varje problem vi stött på och inte kunnat finna lösningar till, kände vi hur vår motivationsnivå sjönk tills vi nått den nivå då vi inte ville arbeta längre. Det fanns många faktorer som spelade roll. Det började med att vi inte fick rotationerna att fungera, detta tog alldeles för lång tid att lösa. Sedan gick det över till att vi inte kunde läsa in våra egna shaders i motorn. Tillsammans med dessa stora problem uppstod även mindre problem, som vi av tidigare erfarenheter visste skulle fungera. Dessa oförklarliga problem löste sig ibland genom att skriva om exakt samma rad kod på precis samma sätt. Detta saknar helt logisk förklaring.

En annan bidragande faktor var bristen på lärare som var kunniga inom spelprogrammering, då Jonas Svegländ var tjänstledig under denna tid. När vi hade problem med rotering och rendering så fanns det ingen hjälp att tillgå från högskolan.

Vi var också allmänt trötta på utbildningen, då vi tidigare läst irrelevanta kurser som inte hade med spelprogrammering att göra. Det saknades även kurser för att tillgodose kompetens för de roller som krävs för att göra ett spel.

De verktyg som användes för utvecklingen av spelet mot spelkonsolen kändes ofärdiga och på så sätt förhindrade det oss att förverkliga vår vision av det färdiga spelet, vilket var ett stort nederlag för oss.

I början så bet vi ihop tänderna och försökte ändå att arbeta, dock så fick vi inte så mycket gjort som vi hade önskat. Allt eftersom tiden gick utan att vi hade några större framsteg sjönk motivationsnivå så pass mycket att vi inte kunde arbeta mer. Vi tog en veckas paus i hopp om att få nya krafter för att återigen kunna fokusera på arbetet. Detta hjälpte inte och vi låg nu flera veckor efter i planeringen och kände ångest över att inte kunna få något som var spelbart klart i tid.

Slutfasen

Genombrottet för oss blev när vi beslutade oss för att byta miljö. Vi tömde projektrummet i högskolan och förflyttade allt till Joakims lägenhet. Plötsligt så fick vi nya krafter och projektet tog fart på allvar. Alla små problem som vi haft orkade vi nu rätta till och kunde sedan koncentrera oss på spelets gameplay.

Äntligen så kunde vi skjuta ner fiender och även använda skeppets sköld för att absorbera dem. Nu när allt grundläggande gameplay fungerade så testspelade vi en del. Vi var nöjda med det och tyckte att spelet var roligt. Det vi däremot inte var helt nöjda med var utseendet på spelarskeppet och omgivningen som man rörde sig i. Vi pratade igenom det och bestämde oss för att spelarmodellen skulle var mer abstrakt. Joakim ritade några skisser och blev mest nöjd med en skiss där spelarskeppet bestod utav tre ringar som roterar runt sin egen axel och i centrum av detta finns en boll av ren energi.

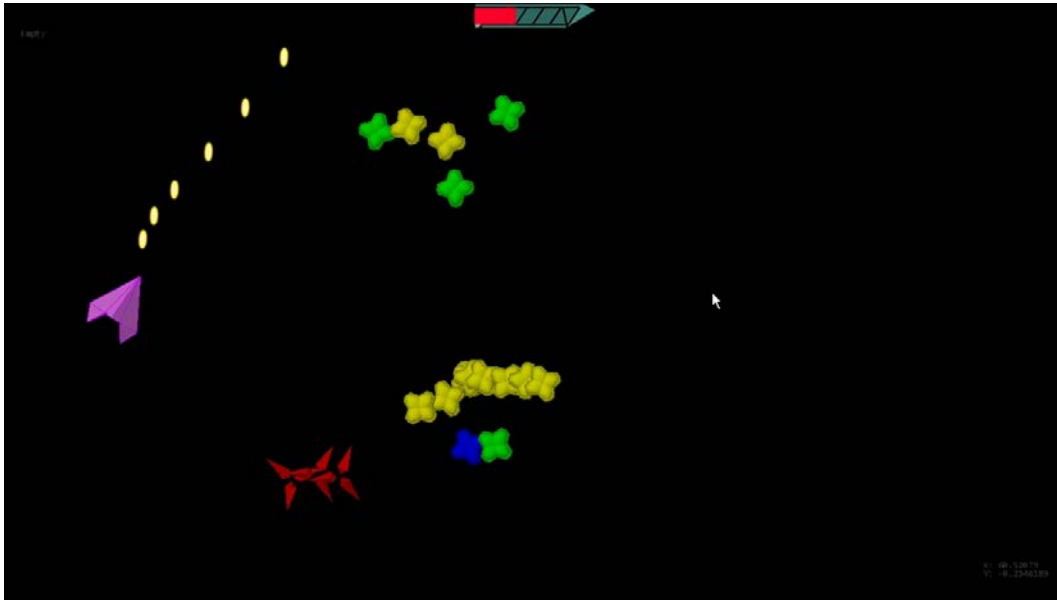


Detta är resultatet efter omdesignen av spelarskeppet, dock visas inte partikelsystemet i dess mitt.

Vi programmerade partikelsystem som vi lade in för att få effekter som explosioner när spelaren skjuter ner fienden. Ett partikelsystem är ett system som har hand om att skapa partiklar, rita ut dem och ta bort dem efter att de dött. Varje partikel har en livslängd som bestäms när den skapas. En partikel kan se ut som vad som helst, det kan vara en bild eller en 3d-modell. I vårt fall bestod de av en bild som föreställde en "boll" av ljus. Ett

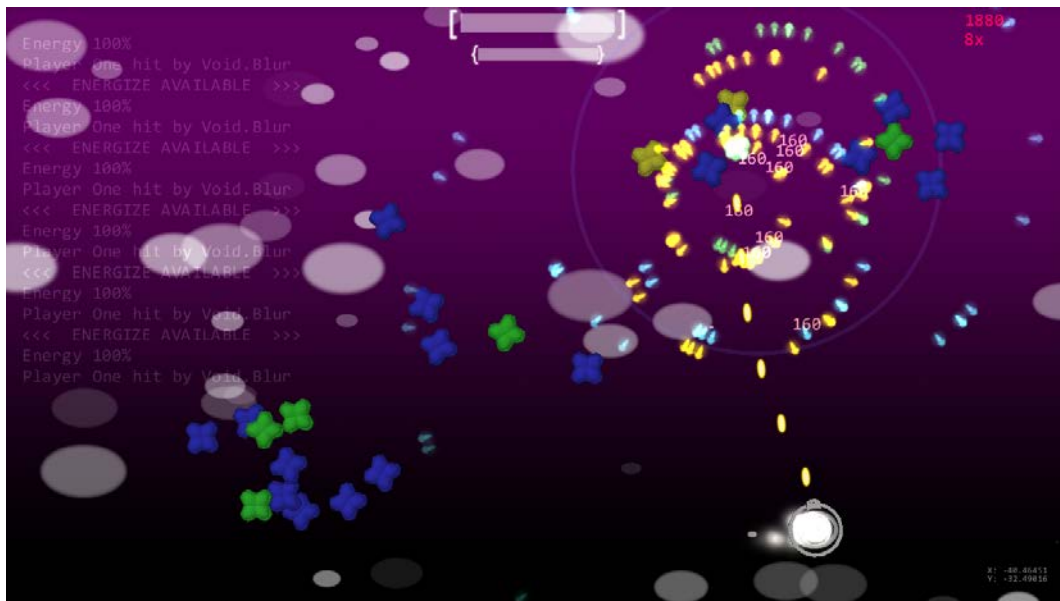
partikelsystem användes även för att skapa effekten av spelarens energiboll.

En annan sak vi också valde att designa om var spelets HUD. Vår dåvarande HUD visade bara spelarens energi och grafiken var endast temporär och fyllde enbart en praktisk funktion så att vi skulle se hur mycket hälsa som fanns. Den nya grafiska designen vi skapade hade både energimätare och en mätare för spelarens sköld. Vi implementerade även en funktion som ritade ut en bakgrund med färger som växlade under spelets gång.



Ovan ser man hur spelet såg ut innan vi fick tillbaka motivationen och gjorde en omdesign. Det fanns inget gameplay då skotten man sköt inte kunde skada fienderna och man fick inga poäng. Bakgrunden var helt svart och det fanns inga partikelsystem.

Nedan ser vi hur långt vi kom på vår prototyp efter att vi bytte arbetsmiljö. Spelarmodellen har helt bytt utseende och det skapas partikelsystem när fienderna dör. HUD'en har även den fått ett ansiktslyft. Nu visas både spelarens energi- och sköldmätare, spelarens sammanlagda poäng visas uppe i högra hörnet och under den kan man se sin kombopoängsvariabel. Som vi även kan se är bakgrunden inte längre enfärgad svart utan skiftar färg konstant samtidigt som semitransparenta "moln" passerar förbi.



Slutord

Det har varit en svår väg att ta sig fram till den slutprodukt vi har skapat. Under den största motivationssvackan trodde vi inte att vi skulle få fram en bra prototyp av vår spelidé. Men om man ser till hur långt vi har kommit känns det ändå som att vi åstadkommit mycket av den vision vi hade från början. Den största besvikelsen var att vi inte lyckades ge musiken den roll vi från början önskade. Men med de idéer vi lyckats förverkliga, och den prototyp dessa resulterade i, känner vi oss ändå nöjda. Vi anser att spelet håller Xbox Live Arcade standard och tycker själva att det är roligare än många av de spel som vi spelat där. Med största sannolikhet så kommer vi att fortsätta arbeta på detta spel även efter att examensarbetet avslutats och då försöka få in allt det som vi inte hann få med. Vi vill fullfölja den vision som vi hade från början och hoppas att spelet en dag går att finna på marknaden.

Bilagor

Projektplan XNA spel

jolp05 frga05 bmni05

Målet med projektet:

Att efter avslutad kurs ha en spelbar prototyp av det XNA spel vi hade tänkt utveckla. Vårt mål är att spelet ska vara tillräckligt bra för att kunna vidareutvecklas till ett färdigt XBLA (Xbox Live Arcade) -spel. Spelet ska kunna platsa bland de övriga titlar som finns ute på Xbox Marketplace.

Syftet med projektet:

Att få insikt och kunskap hur det är att programmera ett spel till en annan plattform än PC, i detta fall Xbox360.

Projektmetod:

För att kunna uppnå ovan nämnda mål använder vi oss av Communitys som finns runt XNA - <http://creators.xna.com/> - samt den mjukvaruutvecklingsmiljö som finns att tillgå när man köpt en licens på Xbox Live Market Place via Xbox360. Detta avser vi att göra.

Tidsplan för projektet:

Vecka 4: Skriva ett designdokument samt införskaffa licens för Creators Club och lära oss allt om detta.

Vecka 5: Gå igenom guider och den information som erbjuds via Creators club och även bekanta oss med C#, vilket är ett nytt programmeringsspråk för oss.

Vecka 6: Klarlägga arkitekturen för spelmotorn, skapa klassdiagram, flödes diagram m.m. Arbetet kommer här att brytas ner i konkreta implementationsbara uppgifter samt delegeras ut till ansvarig för respektive uppgift.

Vecka 7: Programmering

Vecka 8: Mot slutet av denna vecka ska en spelbar prototyp finnas för test av spelbarhet. Även en skiss på spelmusik ska finnas färdig för utprovning.

Vecka 9 - 13: Programmering samt ljudinspelning.

Vecka 14: Produktion av 3D-modeller och texturer.

Vecka 15 -17: Kvalitetskontroll och utprovning.

Vecka 18: Examination.

Eventuell ekonomi för projektet:

Två Creators Club medlemskap à 900 kronor.

Behov:

Ett eget projektrum som arbetsplats samt för säker förvaring av den utrustning vi behöver.

Risker med projektet:

Brist på motivation kan bli det största problemet. En lösning på detta problem kan vara att titta på redan befintliga XNA-Spel och söka inspiration i dem.

Genom programmeringen kommer vi att stöta på många svårlösta problem vilka kan rubba vår planering. XNA och C# kan bjuda på lite motstånd då vi varken har programmerat med varken XNA eller C# tidigare. Vi kommer att försöka lösa dessa problem genom att söka svar i böcker och/eller på nätet.

Övrigt:

Designdokumentet, som ska skrivas under första veckan, kommer att fungera likt en bilaga till projektplanen. Designdokumentet kommer att innehålla information om vår vision av hur spelet ska se ut samt fungera; hur man interagerar med spelet, ens mål samt de regler som finns för att kunna nå detta mål. Det kommer med all sannolikhet att uppdateras under projektets gång allt eftersom vi inser vad som fungerar eller inte fungerar.

Ett tekniskt dokument, som först kommer att vara en övergripande blick av spelmotorn. Detta dokument kommer att inkludera kod och alla klasser m.m. och kommer redovisas genom klass diagram och flödesscheman. Den slutgiltiga versionen kommer att utgöras av ett Doxygen-genererat dokument, som ska motsvara den exakta uppbyggnaden av spelmotorn och dess arkitektur.

Personlig reflektion över projektarbetet "Void" av Joakim

Inledning

Anledningen till att jag ville arbeta tillsammans med Bo Martin och Fredrik var att vi har arbetat tillsammans i tidigare projekt så vi visste alla var vi hade varandra och hur mycket vi kunde.

Jag har under en längre period varit intresserad av att programmera ett spel för konsol istället för PC, vilket alla tidigare projekt har varit. Microsoft Xbox360 och XNA var det mest självklara valet då ingen annan konsol har något officiellt utvecklingsverktyg för icke licensierade utvecklare. Dock så medförde det att vi blev tvungna att lära oss ett nytt programmeringsspråk, nämligen C#. Tidigare har jag bara jobbat i C++ och det är det enda programmeringsspråk som vi har läst. Jag trodde inte att det skulle vara så svårt att lära sig då C# bygger på C++ och det var det inte heller men skillnaden i syntax var lite större än jag trott. Den absoluta skillnaden är dock hur det hantera allokering och deallokering av minnet. Då C++ är fritt att använda som programmeraren önskar finns det vissa restriktioner i C#. C# använder sig nämligen av garbage collection (GC) vilket innebär att datorn "själv" bestämmer när den vill deallokera den del av minnet som inte används längre. Detta kändes inte bara skrämmande utan kan även påverka spelets prestanda negativt, den positiva delen består i att spelet inte kan skapa minnesläckor.

Mitt bidrag till projektet

Det jag helst ville arbeta med i projektet var High Level Shading Language (HLSL) programmering även kallat shader programmering som är en mer generell term. Det är detta som bestämmer hur objekt ska ritas ut på skärmen, och det är genom detta man kan skapa olika effekter, t.ex. att få ett material att se plastiga ut eller att få något att reflektera sin omgivning likt en spegel. I dagsläget används shaders i praktiskt taget alla spel som finns ute på marknaden och det är dessa som bestämmer hur bra ett spel ser ut grafiskt. Dock så fanns det inte tillräckligt med tid för mig att fördjupa mig så mycket som jag önskat i ämnet, då det fanns mycket annat som var tvunget att göras först. Tyvärr så finns det inte så många programmerare på skolan och vi var bara två stycken i detta projekt så vi fick båda göra saker som vi inte var särskilt intresserade av.

De delar som jag främst tog hand om i spelmotor var

- Spelmekanik
- Kollisionssystem
- Spelarens vapen
- System för att läsa in XML filer som sedan styr skapandet av banor och fiender
- 3dmodeller och texturer
- Shaders

men det finns även andra arbetsuppgifter som både jag och Fredrik samarbetade med

Problem och motgångar

De största problem som uppstått rent kodmässigt var problem med att rotera objekt och att kunna köra våra egna shaders. Av någon anledning så blev alla de 3dmodeller som vi använde oss av, felroterade och om vi försökte roterade dem rätt i vårt spel så försvann rotationen runt en annan axel. Detta tog lång tid att lösa och kan läsas mer om i veckorapporterna.

Ett annat var att kunna läsa in våra egna shaders genom XNA. Det var inte våra shaders som det var fel på, utan vi hade problem med att komma ifrån det redan befintliga shaders som fanns inbyggda i XNA och istället köra våra egna, men detta löste sig också till slut.

Sen har det även funnits en massa mindre problem och buggar som har tagit upp en stor del av vår tid, men det absolut största problemet var att vi efter en viss tid in i projektet helt tappade vår arbetsmotivation. Anledningen till att vi tappade vår arbetslust var bl.a. på grund av de problem vi stött på i projektet, att det tagit sådan lång tid att lösa och att det inte funnits någon lärare med den kompetens som behövts för att kunna hjälpa oss.

Detta blandat med att jag under en längre tid mått dåligt på grund av att denna utbildning känts ofullständig. Den känns inte alls genomtänkt då det fattas nödvändig kompetens för att kunna utveckla spel.

Ett lysande exempel på detta är avsaknaden av en grafikerinriktning, då vi bara har speldesign och spelprogrammering. En kurs som vi läste hade som mål att motsvara en skarp spelproduktion och för att kunna göra detta krävs även grafiker och inte bara programmerare och designers, vilket den dåvarande programansvarige helt verkade strunta i, trots kritik från studenter. Bristen på grafiker återspeglas även i vårt projekt, då jag som programmerare även har fått ta på mig rollen som grafiker, detta utan någon som helst utbildning i ämnet.

Många kurser som vi läst har känts helt irrelevanta för utbildningen och skulle kunnat ha bytts ut mot mer lämpliga kurser.

Slutresultat

Om man ser på hur spelet ser ut i nuläget måste jag ändå säga att jag är ganska nöjd med resultatet med tanke på hur lång tid som gick bort på grund av motivationsproblemen. Tyvärr kunde vi inte slutföra den vision som vi hade från början då, XNA satte käppar i hjulet för oss. XNAs system för att hantera ljud och musik var inte så utvecklat som man kunde ha önskat. Om jag skulle ha gjort något annorlunda skulle det vara hur vi planerade arbetet. Istället för att bestämma vad som skulle göras varje vecka så skulle jag bestämma varje morgon vad som skulle göras den dagen. Detta var något som vi gjorde i slutet av projektet och det kändes som vi fick mycket mer gjort på detta sätt. Listan av saker som måste göras blev inte lika skrämmande om man bara satte upp några punkter per dag istället för massor under en hel vecka och det blev lättare att fokusera sig.

Personlig reflektion över projektarbetet "Void" av Fredrik

Inledning

Arbetet med spelet "Void" påbörjades i vintras och är nu många veckor senare i så gott som spelbart skick.

Valet av arbetsgrupp var lätt, jag ville arbeta ihop med Joakim och Bosse då vi gjort spel ihop tidigare, och vi vet hur bra vi jobbar tillsammans. Jag och Joakim har även jobbat tillsammans med olika projekt under hela utbildningen och har ett mycket bra samarbete.

Jag tycker att vi gjorde rätt i att utveckla vårt spel i XNA så att vi fick en annorlunda utvecklingsmiljö att arbeta med. Vi var även tvungna att lära oss C# för att använda XNA som även det var kul att testa på. Det var spännande och ganska kul att få lära sig ett nytt programmeringsspråk, dock så var inlärningsperioden för att lära sig C# längre än beräknat. Jag trodde inte att C++, som var det språk jag kan sedan tidigare, skulle skilja sig så mycket från C#. Den största skillnaden var minneshanteringen, då C# har själv koll på när minnet skulle allokeras eller deallokeras. Detta är en allt för krävande process och jag anser att man inte bör använda C# för spelprogrammering. Dessvärre var C# det enda alternativ vi hade, för att kunna utveckla mot Xbox360.

Mitt bidrag till projektet

Att få programmera gameplay har alltid varit mitt största intresse inom spelprogrammering, t.ex. att få fienderna att röra sig på ett speciellt sätt, eller att skapa ett system som håller koll på hur mycket hälsa, ammunition eller vilka vapen spelaren har. Så detta blev min roll i vårt projekt. Jag tog hand om fienderna och fick dem att röra sig och anfälla, gjorde ett menysystem samt en HUD. Nu fanns det tyvärr inte så många programmerare i vår grupp så jag och Joakim fick även bygga upp spelmotorn tillsammans innan vi kunde sätta oss med de delar vi ville arbeta med. Detta tär på arbetslusten att inte få jobba med det man vill utan blir tvungen att programmera allt runt omkring först för att sedan kunna programmera t.ex. spelarens styrsystem. Även fast vi hade olika arbetsuppgifter så satt vi ibland tillsammans och hjälptes åt för att underlätta programmeringen.

Problem som uppstod under arbetets gång

Problemen som uppstod under arbetets gång var många och svårlösta. För det första så tog det längre tid än beräknat att lära sig programmet XNA (som behövdes för att kunna programmera mot Xbox) då syntaxen var skiljde sig från DirectX 9.0c (2002, Microsoft) för PC som jag hade erfarenhet från tidigare. Xbox360 har en modifierad version av DirectX9.0c som XNA använder sig av. DirectX är en samling API som har hand om t.ex. ljud, grafik och inmatning via tangentbordet eller musen.

Ett annat problem som jag stötte på var att få rotationerna av objekt att fungera i den spelmotor som vi hade byggt upp. Av någon anledning så ville inte modellerna jag läste in roteras rätt. Jag vet hur man roterar objekt i en 3D-värld, men det blev ändå inte rätt i XNA. Tillslut la jag rotationerna åt sidan för att kunna ta tag i andra viktiga saker som att ha ett fungerande menysystem. Detta gick inte heller så bra då jag försökte göra menysystemet alldeles för avancerat för det ändamålet vi tänkte ha menyerna till. Jag hoppade mellan rotationerna och menysystemet tills jag inte orkade sitta med något av dem. Tyvärr så fanns det ingen lärare eller handledare på högskolan som kunde hjälpa oss med våra problem som rörde grafikprogrammeringen. När vi väl lyckades lösa problemen så uppkom nya problem och jag blev så trött på att programmera att jag knappt orkade göra något på spelet.

Jag var på begravnings mitt i alla problem som omgav oss. Det sänkte min motivation ytterligare, och alla dessa saker som bromsat oss gjorde så att vi var tvungna att ta en paus.

Tankar över slutresultatet

Om jag tittar på hur spelet ser ut nu, efter att vi fick tillbaka våra krafter och gjorde de ändringar vi tyckte behövdes, så är jag nöjd. Det känns som en professionell prototyp av ett spel.

Hade jag hade fått göra om spelet på 15 veckor så hade jag gjort en del saker annorlunda. Jag började oftast för storskaligt på de delar jag hade ansvaret för. Detta slutade oftast med att jag blev tvungen att börja om många gånger. Pågrund av dessa handlingar kom jag efter i planeringen och blev tvungen att jobba under mer stress än vanligt.

Vi kom på ett bra sätt att lägga upp arbetet de sista dagarna i projektet. Det gick lättare att ge oss några uppgifter som var tvungna att lösas varje dag, istället för att planera veckovis. T.ex. att vi skulle programmera en bakgrund som skiftade färg eller att vi skulle göra så att fienderna blinkade när man träffade dem med ett skott. Detta gjorde att vi slapp se den stora högen med arbete och var inte lika stressande som att se en hel veckas planering.

Hade vi använt oss av detta system från början av projektet så tror jag att vi hade kommit lite längre när deadline var nådd.

Personlig reflektion över projektarbetet "Void" av Bo Martin

Inledning

När jag sökte till denna utbildning hade jag som mål att utveckla mitt musikskapande. Jag hade också som mål att lära mig använda datorn som ett verktyg för hela processen i ett musikskapande, från t.ex. en spontan melodi i huvudet, till ett färdigt arrangemang. Tidigare hade jag enbart använt portabla studios, där inspelning av musikspår skedde direkt och med färre möjligheter till effekter och redigering i efterhand. Genom att använda en dator med samplade instrument har man t.ex. mycket större möjlighet till efterredigering och korrigerings.

Mycket har också hänt rent tekniskt under de tre åren jag gått på skolan. Verktygen för hemmastudios har blivit bättre och billigare vilket gjort fullt möjligt att på egen hand genomföra professionella produktioner. Utbildningen har på ett bra sätt beaktat detta vilket gjort att vi haft möjlighet att jobba med mycket modern teknik. Den nya studion är ett bra exempel på detta.

Utbildningens innehåll

Innehållsmässigt har utbildningen täckt mycket av det man hade som förväntan. Till exempel har vi fått göra radioteater, mixningar, komponering för orkester, ljudläggning av film, programmering av ljudmotorer för digitala medier för att nämna några. Ofta kan det ha känts att vi bara skrapat lite på ytan, men tredje året gav oss en chans att fördjupa oss i något av det vi tidigare arbetat med, som vi fann extra intressant.

En del önskningar över

Som en del av första kullen i Digital Ljudproduktion har det också känts att programmet varit helt nytt. Ett fåtal kurser har dessvärre känts rakt igenom meningslösa. Till exempel hade vi en kurs vid namn "Case kurs" där vi parades ihop med slumpmässigt utvalda studenter från de andra programmen i syfte att ta fram en produkt eller idé åt ett antal utvalda företag. Konceptet låter kanske alldeles vansinnigt lysande, men rent praktiskt var kursen ett fullkomligt misslyckande. Den passade säkert jättebra till programmet Medieteknik, då många av önskemålen från företagen passade ypperligt med detta programs agenda. I den grupp jag så olyckligt hamnade i bestämdes dock att designa hemsidan åt skyltbolaget 4Sign. Var kommer digital ljudproduktion in under utvecklandet av en webbtjänst? En viktig fråga jag ställde den kursansvarige. Med ett visst medhåll om att frågan var svår att besvara fick ändå, inte helt oväntat svaret att jag ändå skulle försöka komma på något samt att det också var en del av kursen att komma underfund med just detta.

Så här i efterhand hade jag önskat att jag ställt högre krav och bett att få genomföra någon alternativ kurs mer riktad åt ljudproduktionshållet. Eller kanske bara fått byta grupp för det fanns faktiskt de grupper som haft turen att få ett projekt med större möjligheter för en ljudproducent. Mångfalden bland olika möjligheter till projekt gick tråkigt nog före en enskild elevs intresse i detta fall.

Övriga tveksamheter

En annan fullkomligt rykande usel kurs var kursen Soundscape. Den berörde, för mig, på ett sätt ljud mer än Case-kursen. Tyvärr höll inte läraren måttet riktigt. Hon var konstnär och personligen är jag lite väl cynisk för att klara av en konstnärs värderingar. Tråkigt nog delade jag åsikt med många andra i klassen, men inte bara på grund av lärarens något magra sätt att tänka. Hon saknade helt enkelt kunskap att undervisa en klass där ambitionsnivån och det generella kunnandet, vad gäller musik och ljud, låg högre än en lågstadielklass. De saker hon visade var föga imponerande och var snarare ett steg tillbaka till en tid då man experimenterade med allt, utan något direkt mål med det man gjorde; en tid då man enbart skapade utan vetskap om resultatet, ett knappast effektivt arbetssätt vem som helt kan anamma. Det var en svår period då jag och många andra faktiskt hade som funderingar att hoppa av utbildningen.

Slutord

Jag har lärt mig mycket under utbildningens gång. Inte bara hur olika verktyg fungerar, utan den kunskap jag fått under utbildningens gång har också lärt mig kritisera det jag själv skapar. Detta hjälper mig att ständigt prestera bättre.

Överlag känns det som att sista året på utbildningen var det som gav mest, men med den självkritik jag anammat, av lärdom från tidigare kurser, hade det inte varit utan de tidigare kurserna de senare varit de mest givande. Första halvårets kurser kändes kanske lite onödigt, då vi helt delade kurser med medieteknik. Men efter att ha sett klasskamrater, i sina slutprojekt, dra nytta av de kunskaperna vi fick då kan jag inte helt döma ut allt vi lärde oss då. Det kanske känns som onödigt vetande för en ljuddesigner, hur man ljussätter och komponerar fotografier eller filminspelning, men det kan ju faktiskt vara en nyttig kunskap ifall man skulle söka jobb på till exempel Sveriges Television.

En annan kunskap många kände var onödig, och som var en del av första årets kurser, var programmeringen. Den har jag dock haft stor nytta av under hela utbildningen. Inte bara som grundkunskap för bättre förståelse av ljud i spel, utan även praktiskt då jag i två stora projekt har haft nytta av programmeringen. I slutprojektet nyttjade jag inte bara mina programmeringskunskaper utan även kompositionskunskaper och kunskaper jag fått av kursen Tematisk fördjupning.

Ordlista

XNA – Ett utvecklingsverktyg för utveckling av spel mot Microsofts Xbox360.

API – Application Programming Interface, Ett API fungerar som en mellanhand mellan hårdvara och utvecklingsverktyg

C# – Ett objektorienterat programmeringsspråk.

Gameplay – Alla de handlingar man utför i ett spel.

Lead – En benämning av en ledarskapsposition i ett utvecklingsteam.

Källförteckning

Ikaruga (2001) Treasure Co. Ltd. Japan: Sega. (digitalt spel: Sega NAOMI)
(<http://www.xbox.com/sv-SE/games/i/ikarugaxboxlivearcade/>) (2008-05-26)

Space Invaders (1978) Taito Corporation. Japan: Midway. (digitalt spel: arkad)
(http://en.wikipedia.org/wiki/Space_Invaders⁶)(2008-05-26)

Geometry Wars (2005) Bizzare Creations. Worldwide: Microsoft (digitalt spel: Microsoft Xbox360)
(http://www.bizarrecreations.com/games/geometry_wars_retro_evolved) (2008-05-26)

Mutant Storm Reloaded (2005) Pom Pom Games. Worldwide: Microsoft (digitalt spel: Microsoft Xbox360)
(<http://www.xbox.com/en-US/games/m/mutantstormreloadedxbox360/default.htm>) (2008-05-26)

Rez (2001) Sega. Japan: Sega (digitalt spel: Sega Dreamcast)
(<http://www.sonicteam.com/rez/e/news/index.html>) (2008-05-26)

Boom Boom Rocket (2007) Bizzare Creations. Worldwide: Electronic Arts (digitalt spel: Microsoft Xbox360)
(<http://www.bizarrecreations.com/games/boom%5Fboom%5Frocket/>)(2008-05-26)

Harris, A.(2002). *C# programming for absolute beginners*. Boston: Course PTR.

Ageia PhysX –Heter numera det Nvidia PhysX
(http://www.nvidia.com/object/nvidia_physx.html) (2008-05-26)

DirectX
([http://msdn.microsoft.com/sv-se/directx/default\(en-us\).aspx](http://msdn.microsoft.com/sv-se/directx/default(en-us).aspx)) (2008-05-26)

Visual studio
([http://msdn.microsoft.com/sv-se/vstudio/products/default\(en-us\).aspx](http://msdn.microsoft.com/sv-se/vstudio/products/default(en-us).aspx)) (2008-05-26)

XNA - <http://creators.xna.com/>(2008-05-26)

XAct - <http://creators.xna.com/>(2008-05-26)
Xact är en del av XNA

Renoise(2000, Eduard Mueller och Zvonko Tesic)
(<http://www.renoise.com/>)(2008-05-26)

⁶ Detta är ingen säker källa, men då spelet är så gammalt så har det ingen officiell hemsida