Projekt “the adventures of some weird place”
GRUPP 4

Kurs: Projektkurs inom Datateknik och internetteknik 9.0 hp

Titel: the adventures of some wierd place

Grupp: 4

Deltagare:
Lucas Persson,
Ivan Berezkin,
Marek Hudzik,
Julia Thun,
Stefan Tuiskunen

Handledare: Reine Bergström

Examinator: Anders Lindström

Datum: 2013 05 16

Sammanfattning

Projektgrupp 4 fick uppgift att skapa ett nätverkbaserat spel. Gruppen bestämde sig för att göra ett samarbetsplattformspel för Linux, Mac och Windows. Spelet skulle bestå av olika nivåer, där det fanns olika hinder och problem man skulle lösa.

Gruppen bestämde sig för att programmera i OpenGL, c++ och c. Gruppen fick även en virtuell server på KTH som vi kunde ansluta klienter till och få det att fungera genom nätverk.

Gruppen använde c++ i den grafiska delen eftersom det är enklare att utveckla ett spel när det är objektorienterat. Att skapa klasser för players, boxar, plattformar och så vidare hjälper en hel del.

Gruppen fick även rita figurer som skulle föreställa spelare, bakgrundsbild och andra effekter som vi senare skulle föra över genom kod till spelskärmen.

Projektgruppen lyckades sätta ihop en kommunikation mellan klienterna och serven, rita upp det grafiska i spelet, föra över det till skärmen och göra klart en nivå med problem och hinder. Där två olika spelare kunde sitta och spela tillsammans online .

Summary

Projectgroup 4 was tasked with creating a network based game. The group decided to do a cooperative platform game for linux mac and windows. The game would consist of different levels, each level would consist of various obstacles and problems that the players had to solve in order to move on.

The group decided to program in OpenGL, C and C++, because its object oriented and easier to develop a game with classes and object when you use the same code over and over. The group was granted a server at KTH, we were to establish a connection between this server and the client.

The group also drew their own figures that would represent a player, wallpaper and other effects that we could use. After that we had to get those picture ”in to the screen” through an algoritm.

The project group managed to establish a communication between the clients and the servar, draw the graphic in the game, “transfer” it to the screen and make a whole level with problems and obstacles that two different players could play together online.

Förord

Den här rapporten handlar om programvaruteknik, spelutveckling. Detta projekt har bidragit till större kunskap inom områden såsom objektorienterad programmering (C++), grafik (SDL, OpenGL) och nätverksprogrammering (client/server modell).
Till sist ett tack till universtitetsadjunkt och mentorn Reine Bergström för stödet du har givit under detta projekt.

Innehåll

Inledning

Projektgruppen fick i uppgift att skapa ett nätverksbaserat spel. De få kraven som fanns på projektet är:

- Det ska gå att spela tillsammans med en annan klient genom en servar.

- Gruppen ska ha ett fungerande spel när det presenteras.

Resten fick gruppen bestämma själva, hur "storyn" skulle vara, hur animationen skulle se ut, hur grafiken ska se ut osv. De största problemen är att gruppen inte har så mycket kunskap inom tillverkning av spel och att gruppen inte har ritat grafiska föremål på datorn förut(fast de har vi). Målet är att skapa ett fungerande spel där man behöver samarbeta för att kunna ta sig vidare i spelet. Skapa en spelmotor och rita ut grafiken i spelet, få mer kunskap bl.a. inom C++, grafikritning, SDL/OpenGL, nätverk och hur man arbetar inom programmerings projekt. Gruppen har använt sig av tutorials, litteratur och tidigare kunskaper. Tutorials är ett bra sätt att lära sig, då man får göra sakerna själv och får allting förklarat för sig. Detta kan man sedan använda i sin egen kod.

1. Processen

Projektgruppen hade som sagt stor frihet när det gällde hur spelet skulle se ut, fungera osv. Det första gruppen gjorde efter att ha fått uppgiften var att samla ihop alla deltagare och diskuterat de största delarna av projektet dvs.

1.1 Typ av spel

Gruppen pendlade mellan att göra en top-down first person shooter och ett plattformspel. Men efter att ha vägt upp alla för- och nackdelar så bestämmde gruppen sig för ett plattformspel bl.a för att det är lite svårare (Gruppen ville utmana sig själva) och det är mer intressant att skapa ett plattformspel.

1.2 Story, problem och andra detaljer

Gruppen var tvungen att komma på en story till spelet och bestämma vad spelet skulle gå ut på. Gruppen valde att göra ett samarbets spel där man var tvungen att hjälpa varandra för att kunna ta sig vidare i spelet. Exempel på problem är att en spelare är tvungen att ställa sig på en stenblock så att den andra spelaren kunde ta sig vidare och öppna "en dörr". Sen var gruppen tvungen att bestämma hur grafiken skulle se ut. Gruppen valde att rita upp sin egen grafik och sedan "ladda in" det i spelet, det skulle både vara roligare att göra eget samtidigt som vi hade fått bra kunskaper då ingen i gruppen hade gjort det förut.

1.3 Uppdelning av arbetet

Projektgruppen skulle bestämma vem som gör vad. Där valde gruppen att dela ut små uppgifter till enskilda personer t.ex. nån skulle rita gubbar, nån annan bakgrund osv. De stora uppgifterna som att sätta upp kommunikationen och skriva koden fick alla samarbeta med varandra. Deltagarna i gruppen arbetade mycket enskilt, men man hade möten ungefär tre gånger i veckan då man kunde träffas och gå igenom vad man har gjort, samtidigt som man kunde berätta om sina problem och förhoppningsvis få några tips på lösningen. Gruppen har självklart haft hel dagar då den suttit och arbetat tillsammans i skolan.

2.Visuellt

2.1 Ritning av grafiken

Rita grafiken var en svår uppgift, då ingen i gruppen hade gjort det förut. Men det funkade bra och vi lyckades framställa några bilder som vi sedan kunde använda i spelet. Det två största sakerna i grafiken var figurer som skulle föreställa spelaren och bakgrunden.

2.1.1 Ritning av gubbarna

Gruppen bestämde sig i storyn att rita två olika figuren, den ena skulle vara manlig huvudperson och den andra kvinnlig medhjälpare (Andra spelaren). Man använde Photoshop och (Julia lägg till din Sketgrejjs)

2.1.2 Ritning av bakgrund

Bakgrunden för vår nivå skulle vara en grotta. Den ritades i photoshop. Man ritade två olika "frames" (bilder) som loopade i bakgrunden så att det aldrig skulle ta slut så länge man var på den nivån. För att man skulle kunna ladda in bilden i spelet så var man tvungen att spara den som bmp-fil och sedan i programmet "*Gimp*" fixa inställningar så att den skulle kunna fungera. Fast sedan så ändrade vi till png-fil istället för att kunna använda transparent bild för spelarens figur och för att slippa använda sdl's inmatning av bilder. Nu används OpenGL för inmatning av bilder och alla plattskärmar laddar nu in bilderna korrekt (sdl-användning är svårt för mac). Detta ledde till att vi fick modifiera algoritmen, men det gick bra och resultatet blev bättre än vid användning av bmp-fil.

2.1.3 Ritning av banan

Ritningen av banan gjordes i själva koden, gruppen skapade en funktion som ritade ut plattformar på skärmen. Det tar lite längre tid att skriva in koden för alla plattformar, men fördelen är att man kan modifiera alla plattformer samtidigt, t.ex. med vår kollision funktion som gjorde att man stannade upp så fort man rörde vid "plattformen". Gruppen ritade även ut andra föremål som flyttbara stenblock, portal och andra små detaljer.

2.2 Programmering av det grafiska

I programmeringen av det grafiska ingår massa olika koder. T.ex. spelarens egenskaper, banans egenskaper och det fysiska motorn (gravitationen, hoppstyrkan, hastigheten osv.). Man använde sig av objektorienterad programmering för att dela upp allting och göra det enklare för sig själv.

2.2.1 Spelarens egenskaper

Varje spelare hade sin egen koordinat som ändrades beroende på spelarens position. Positionen ändrades genom funktionerna "KEY_DOWN och KEY_UP" och tangenterna "A","D" eller höger- och vänsterpil som förflyttade spelaren i x-axeln Samma funktion användes för att förflytta spelaren i y-axeln, dvs. hoppa, fast man använde sig av tangenten "SPACE". Hastigheten gavs också där och kunde modifieras med samma funktion och tangenten "SHIFT", detta gjorde att figuren rörde sig med snabbare hastighet. Gruppen hade även fixat funktionen för dubbelhopp. I dubbelhopp använde man sig av sant/falskt, när man gjorde första hoppet så var det "sant" att man kunde köra hopp-funktionen igen och efter den hade körts för andra gången så ändrades det till "falskt" vilket ledde till att man inte kunde hoppa en tredje gång.

2.2.2 Fysik motorn

Gruppen har skapat en "gravitation" i spelet, så att det ser realistiskt ut när man hoppar så drar den ner en på marken igen. Sen har gruppen fixat en kollision funktion som känner av ifall en spelare vidrör en plattform eller nått annat föremål. Om en plattform blir vidrörd så kommer spelaren stanna där, om en portal blir vidrörd så kommer man teleporteras till positionen för portalens utgång eller om en stenblock blir vidrörd så kommer man flytta den i den riktningen man rör sig. Stenblocken blir också påverkade av gravitationen, dvs. om man puttar ut de från en höjd så kommer de att ramla ner till grundnivån (marken).

2.2.3 Kameran

Först skapar gruppens program ett fönster där spelet ska finnas. Sen finns det en funktion som heter “ChangeSize” som ändrar storleken och gör det proportionerligt för alla plattformar och deras storlek, så att hela spelet kan få plats i rutan som skapas vid program start. Det finns en funktion som heter “renderScene” denna funktion har några små funktioner i sig, bl.a. update som uppdaterar bilden. “renderScene” skapar även en kamera som följer efter spelaren dvs, bilden förflyttar sig beroende på spelarens position. I “renderScene” finns det även “mainDraw” som ritar ut all grafik.

2.2.4 Ladda in bilden

För det har gruppen fixat en funktion som heter “loadImage”. Funktionen börjar med att bestämma vilken typ av png-fil bilden är, sedan letar den reda på filen, öppnar den, läser av den och sparar den i en variabel som senare används i “renderScene” där man framkallar bilden till skärmen.

3. Nätverk

3.1 Servar

Servern är skriven i C och använder sig ett egenskrivet bibliotek “udp.c” där man har tagit hjälp utav en “guide to network programming” för att få anslutning över tcp/upd protokollen. Servern är utformad att den startas och under initieringen så öppnar den en “socket” för nya anslutningar från klienter. Sen för varje anslutning så startar den en ny “tråd” som kommer att hantera anslutningarna med just denna klienten, och i dessa “trådar” är det en loop som tar emot spelarens position eller annan information och lägger informationen i en “global struct” (vilket alla trådar kommer åt) och detta görs då för alla anslutningar och sedan distribueras all information i denna “globala structen” till alla klienter som har anslutit och har sin egen tråd, och för att detta är en loop så gör detta kontinuerligt och sen avslutas tråden om den skulle märka att en spelare inte längre svarar.

3.2 Klient

Klientdelen i spelet är ju den som ritar ut spelet på skärmen men även den delen som skickar och tar emot nätverks trafik med servern och detta har uppnåtts igenom att använda “udp.c” och även implementerat en tråd vilket tjänade i syfte att inte dra ner på FPS:en när anslutingen är långsam, vi behöver även packar ihop och delar upp om flera variabler som berättar för de andra spelarna vart på banan man befinner sig och för det har vi skrivit ett mindre protokoll språk kallar PXYA+ vilket vi uppdaterar ifrån globala structar och får då även enkel kommunikation mellan trådarna.

3.3 Protokoll språket P|X|Y|A|+

För att detta spel ska fungera online så måste klienterna kommunicera med varandra och det uppnås igenom att vi packar ihop variablerna i ett minispråk, detta sker på följande vis:
Playerid | X koordinaten | Y koordinaten | Action | +extra
packas ihop i en textsträng för överföring och sedan delas upp i de andra klienterna.

4. Slutsats

Scrum är ett väldigt bra arbetssätt, då man får arbeta med hela projektet istället för bara en liten del, man “daily scrums” (dagliga möten) där man kan få hjälp ifall något inte går som det ska och man får deltagarna att känna sig att de verkligen är med i projektet. Alla i gruppen jobbade på det sätt som passade de bäst, tutorials, guides, litteratur osv. Detta ledde till att man kunde genomföra uppgifterna som ställdes på en och ge resultat. Gruppen klarade av att sätta upp en kommunikation mellan klienterna och servern och fixa ett fungerande grafiskt spel där två användare kunde sitta på var sin dator och spela “online” genom servern på KTH. Gruppen har gjort det största delen av arbetet, en fungerande nivå. Därför kan man efteråt lägga ner tid på att skapa flera olika nivåer, samtidigt som man kan lägga in flera spelare och effekter. Men allt detta är inget problem då gruppen redan har gjort det och det som behövs för att klara denna uppgift är fantasi. Man kan även utveckla vidare på grafiken, t.ex. göra 3D bilder. Men gruppen klarade av de två kraven som ställdes på projektet:

- Det ska gå att spela tillsammans med en annan klient genom en servar.
- Gruppen ska ha ett fungerande spel när det presenteras.

Referenser

http://beej.us/guide/bgnet/output/html/multipage/index.html
http://www.lysator.liu.se/~tb/pi.50000.html
https://computing.llnl.gov/tutorials/pthreads/
http://www.lighthouse3d.com/tutorials/glut-tutorial/
tutorials av thecplusplusguy ex:
http://www.youtube.com/watch?v=2BsTOgABU1k&list=PL0AB023E769342AFE

Appendix

(bifoga kod, bilder ??)

I appendix placeras det material som är opraktiskt att utnyttja där det egentligen hör hemma i rapporten. Det kan vara tabeller, ritningar, diagram, detaljerade beräkningar, ev. programkod, scheman eller liknande. Hänvisningar till appendix görs i den löpande texten. Appendix placeras sist i rapporten och skall finnas med i innehållsförteckningen. Om appendix blir alltför stort i förhållande till den övriga rapporten, kan det utgöra ett eget häfte. Varje appendix skall ha en liten introduktion.