Projekt TNM061

Gruppmedlemmar:

Jacob Nyman, jacny980
Erik Nilsson, erini048
Lucas Palnén Rung, lucpa286
Daniel Olsson, danol716
Isac Rydén, isary742
Viktor Sandberg, viksa378

The Great Escape - Ett spel skapat i WebGL och Three.js

Beskrivning av projektidé

Idén till projektet kom ifrån ett intresse gemensamt för OpenGL/WebGL som uppstod redan under den första 3D-kursen TNM046. Under den första 3D kursen fick vi lära oss väldigt grundläggande om hur OpenGL fungerade med olika matriser och hur de var tvungna att multipliceras för att få det önskade resultatet. Det kändes mycket omständigt och inte speciellt roligt att arbeta vidare på, men när vi i denna kursen fick möjlighet att testa på WebGL API't THREE.js blev vi genast mer intresserade. I THREE.js kunde vi göra allt det vi gjort i den första kursen med avsevärt mindre kod och på ett mycket enklare och behagligare sätt.

I och med denna nyvunna kärlek till WebGL och THREE.js bestämde vi oss för att försöka skapa ett enklare 3D-spel i miljön. Vår grundidé till spelet var att vi ville skapa allt från grunden för att verkligen lära oss om hur allt hänger ihop och förstå svårigheterna med att skapa ett 3D-spel. Vi kom alla överens tidigt om att spelmotorer som *Unity* och *Unreal Engine* inte var av intresse att arbeta i för att då skulle vi inte få lära oss grunden inom spelprogrammering.

För att hålla spelet enkelt valde vi att det skulle utspela sig i ett rum och att målet med spelet skulle vara att hitta en nyckel för att sedan ta sig vidare till nästa rum. Spelet utspelar sig i ett rum på grund av enkelhet till att designa rum som banor.

Metod och Genomförande

Innan någon kod eller projektet arbetet börjat delades gruppen upp i mindre grupper om två som hade olika ansvarsområden.

En av subgrupperna fick ansvaret för att modellera/animera spelkaraktären/världen. De började med att modellera en karaktär med ett relativt enkelt utseende utan för mycket detaljer då fokuset för projektet var att lära sig alla steg vid spelproduktion utan att grotta ner sig i ett speciellt ämne. När karaktären var klar var det dags att ge den skelettet för att efter det kunna göra animationer. Det var denna delen som tog längst tid för denna grupp. Spelvärlden var relativt enkel att skapa då den byggde mest på boxar och cylindrar och där var det istället texturen som gav objekten liv snarare än formen på objektet som var fallet vid karaktären.

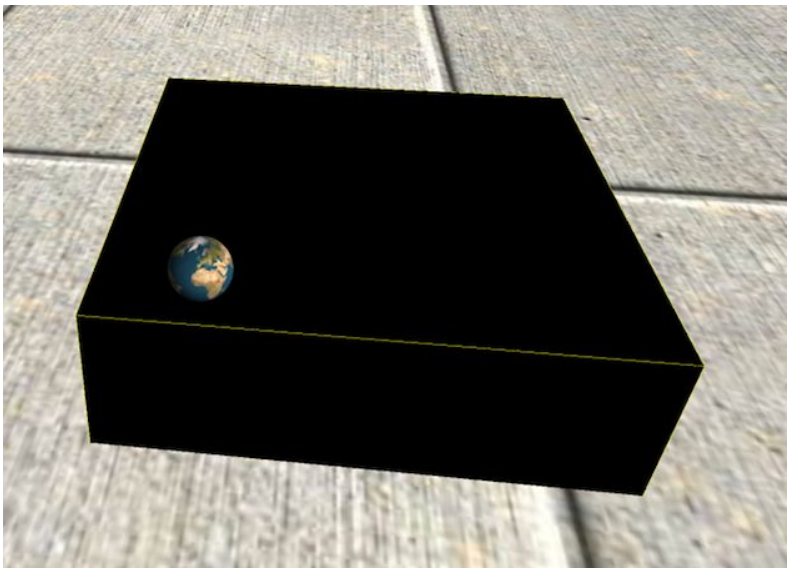


Bild 1 - Tidig bild på scen, när vi fått den grundläggande kollisionen att fungera

De andra två grupperna arbetade med kollisionshantering och karaktärens rörelse. Under arbetsgång insåg vi att karaktärens rörelse gick fortare än väntat att skapa, medan kollisionshantering var den stora utmaningen i projektet. På grund av detta fortsatte en av grupperna med kollisions hanteringen. Kollisionen gjordes på flera olika sätt men tillslut hittades ett sätt som fungerade bättre än de andra, med minst buggar. Den går ut på att ha objekt runt gubben som sedan kollas ifall de kolliderar med andra objekt. Tex, finns det en rektangel framför gubben och ifall den kolliderar med ett objekt så kan inte gubben röra sig framåt. Lite senare in i projektet så fanns det intresse att få in trappor vilket gjorde så att kollisionen fick justeras ytterligare. Det som gjordes då var att dela upp varje objekt till två, så objektet framför fick en övre(vita i bild 2) och en undre del. Ifall den undre delen(röda i bild 2) kolliderade så flyttades gubben uppåt och framåt, vilket gör så gubben kan gå i trappor. Detta fungerar likadant åt alla håll.

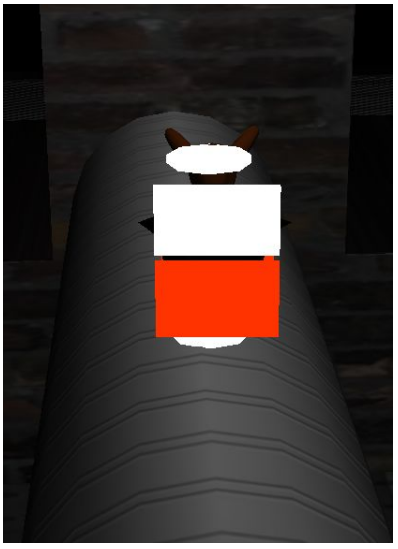
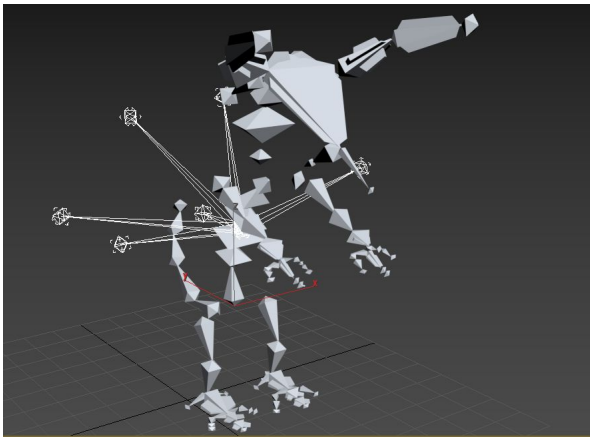


Bild 2 - kollisionsobject, röda för att gå i trappor, vita över för kollision, samt båda om han är i luften

Den andra gruppen fortsatte med att få in modellerna/animationer som modellerar-gruppen skapat in i WebGL. Det var flera problem med att få animationerna att fungera med THREE.js då överföring mellan 3ds max och json gjorde så att längden på animationen ändrades och det gjorde att allt blev svår att styra animationerna.



Huvudkaraktären och dess skelett

För att få spelet att likna ett riktigt spel och inte bara en värld med en karaktär i lades även en del fokus på titelsida, ljud och en meny.
Detta gav i efterhand att spelet blev mer levande och gav en känsla av en färdig produkt.

De verktyg som användes under projektets gång var Github(för delning av kod), Sublime/Notepad++(som texteditor för javascript/html/css), 3DSmax(för modellering/animering). När vi kommit en bit in i projektet och hade en grundläggande kollisionshantering, en karaktär och en bana valde gruppen att fokusera på att få animationer att se bra ut samt att utveckla banan ytterligare med texturer och sist men inte minst att göra kollisionshantering ännu bättre då det fortfarande fanns mycket man kunde jobba vidare på där.

Resultat

Vid projektets slutskede hade ett spel med mycket utvecklingspotential utvecklats. Det finns en färdig bana och nya banor kan lätt importeras och för att utöka spelet omfattning.

En karaktär finns med ett flertal animationer som kan utvecklas vidare och nya kan läggas till utan större besvär.

Kollisionshantering klara majoriteten av hinder och miljöer, men har fortfarande utvecklingspotential.

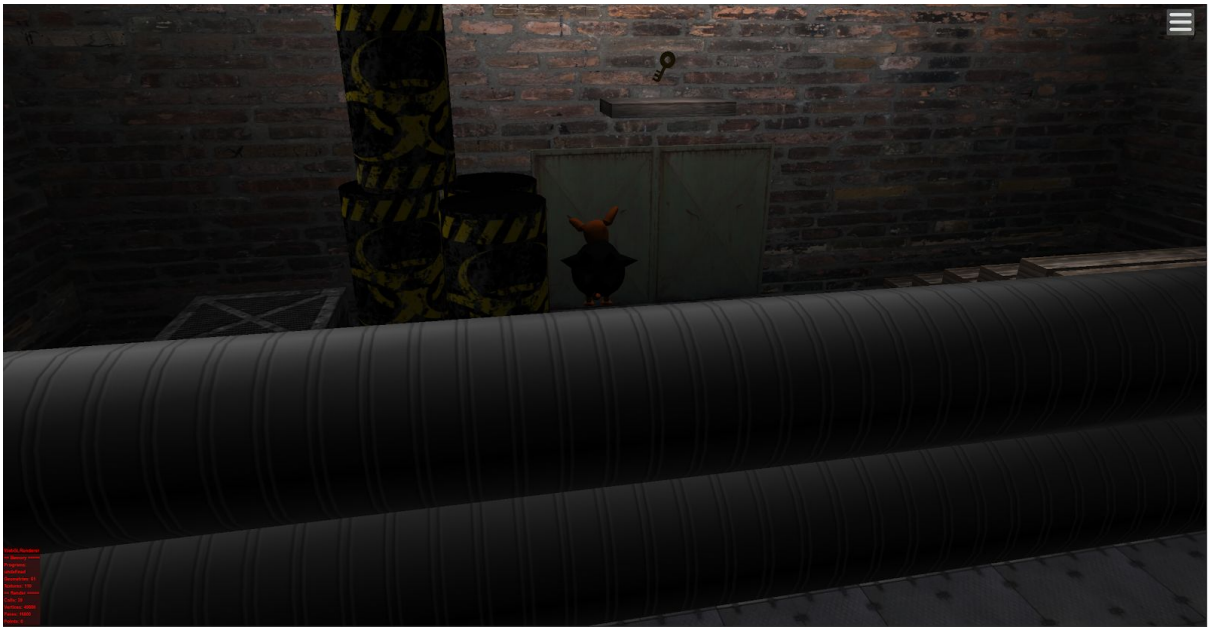


Bild 3 - Färdiga resultatet där fel ljussättning av nyckeln ses

Slutsats och Diskussion

Vi i gruppen tycker att detta har varit ett mycket lärorikt projekt. I och med att vi arbetade med alla delar av att skapa ett spel har vi lärt oss om det som är svårt och det som varit mindre utmanande. Initialt i projekt trodde vi inte att kollisions hanteringen inte skulle bli en större utmaning, men ju mer vi arbetade med det insåg vi att det var relativt svårt att ta hänsyn till alla typer av underlag. Det var också jobbigare att få in texturer och modeller än vad vi trott då vi var tvungna att hitta ett plug-in program till 3DSmax för att kunna exportera modellerna i ett format som vi kunde använda oss av i THREE.js.

Mycket tid lades på att ta reda på vilket format som gick att importera i THREE.js då vi var noviser i detta, först fick gruppen det att fungera med hjälp av obj formatet men det märktes efter ett tag att detta format ej gick att ha animationer i. Många rekomenderade att använda formatet json men då 3ds Max inte kunde exportera till Json valdes detta bort för tillfället. Det tog ett tag innan vi märkte att vi var tvungna att använda JSon formatet om vi inte skrev en egen import klass. Vi fick letat upp en plugg-in till 3DS max som gjorde om till Json men om vi hade vetat från början att vi behövde använda Json så hade vi nog valt att 3d modellera i Blender istället för 3DS max då Blender kunde exportera direkt till Json.

mer info

Avslutningsvis är vi nöjda med resultatet och känner att vi har skapat ett relativt bra spel med tanke på våra begränsade förkunskaper, och tiden vi hade och lägga på detta projekt. Vi känner också att det har väldigt mycket utvecklingspotential så som; nya banor, eventuella fiender, och fortsatt utveckling av andra system.

Hemsidor som används

3ds max till json converterare
http://www.cgdev.net/json/download.php

Json importerare till Three.js
http://www.cgdev.net/blog/482.html

Three.js dokumentation och exempel
https://threejs.org/