PixelJump

ett “Endless runner” spel

skrivet i Java för mobil

Anton Ledström

2014-06-05

Abstrakt

Under 10 veckor har jag utvecklat ett spel för mobil med Android som OS. Spelet är ett "endless runner" uppdelat i olika levlar, till hjälp för att klara detta har du möjlighet att hoppa och volta.

Spelet ska dessutom klaras på en så bra tid som möjligt. Tiden kan du påverka med hjälp av tajming och spelets fysikmotor. Det gäller att hoppa när man får farten med sig vidare.

Spelet är skrivet i Java med hjälp av ett spelbibliotek och en fysikmotor.

Inledning/bakgrund

Målet för mitt projekt har varit att lära mig och testa på att utveckla för mobil, jag tror att jag har stor nytta i framtiden att ha erfarenhet från att utveckla en app för Android.

Spelet är skrivet i Java med hjälp av ett spelbibliotek kallat libGdx och en fysikmotor(Box2D) jag har använt mig av Eclipse som utvecklingsverktyg med Android SDK.

Att testa att utveckla ett spel med hjälp av ett spelbibliotek och en fysikmotor är något jag länge velat testa, en fysikmotor har enorm potential om man kan hantera den.

Tekniker

LibGdx

LibGdx är ett platformsoberoende bibliotek som exporterar för windows, Mac, Linux, Android, IOS, BlackBerry och HTML5 utifrån samma kodbas.

Box2D

Box2D är ett gratis open source 2-dimensionell fysiksimulator motor skriven i C ++
Box2D är en populär motor och används av många spel på marknaden. Box2D finns översatt till många språk bla Java.

TiledMap

Jag har använt mig av tiledmap-editor för att skapa spelets banor, där kan man rita ut spelets banor med hjälp av tiles.
Det går också att skapa fristående objekt och figurer vilket jag använt till att skapa spelets kollisionsdetektion, mellan spelare och “världen”.

Pixel Jump “Endless runner”

Spelet är ett "Endless runner" uppdelat i olika levlar, och till hjälp för att klara spelet har du möjlighet att hoppa och volta. Klarar man en level låser man upp nästa.

Spelet ska dessutom klaras på en så bra tid som möjligt. Tiden kan du påverka med hjälp av tajming och spelets fysikmotor. Det gäller att hoppa när man får farten med sig vidare.

Edit player

Som spelare har man också möjlighet att redigera sin spelkaraktärs utseende, tanken vidare är att spelaren har möjlighet att vinna och låsa upp olika utseenden, tex om man klarar en viss bana eller klarar en banan på en speciellt bra tid.

I spelets nuvarande form har man 3 hattar att välja mellan.

Spelets kontroller

Till vänster om skärmens mitt så hoppar spelaren, till höger gör spelaren en volt. Till höger får spelaren lite mer kraft med sig, dvs gör ett lite högre hopp.

Jag använder mobilens hela skärm som touch.

Positiva Erfarenheter

Övergripande är jag nöjd med mitt projekt och dess struktur, jag tycker att jag har lagt än tämligen god, bred, och genomtänkt plattform att jobba vidare med.

Jag hade en ganska god idé hur jag ville utforma spelet rent objektorienterat, detta från tidigare erfarneheter, boken "Killer Game programming" samt en del youtube tutorials som jag följt tidigare och under projektets gång.
Däremot spelets slutgiltiga spelformat utvecklades fram under projektets gång. Projektet tog en egen riktning vilket jag kände som positivt, jag hade absolut möjlighet att styra projektet i den riktningen jag ville, men det var väldigt spännande att hitta nya saker att implementera och utveckla.

Negativa Erfarenheter

Jag har haft väldigt mycket nytt som jag har behövt lära mig, hur man hanterar att utveckla för en mobilt OS med jars. bibliotek och sökvägar "buildpaths" som inte har fungerat. Box2d och libGdx har också varit väldigt nytt, detta är första gången jag använder dessa plattformar för att utveckla något.

Projektet hade en tämligen trög start där nästan inget ville fungera, stora problem att installera ADT(Android Developer kit) och att få till en emulator för att testa spelet med. Jag skrotade ganska snart att använda emulator eftersom detta tog väldig datorkraft, tid, och tålamod, dessutom var det mycket smidigare att bara kompilera direkt till mobilen.

Att implementera fysikmotorn var något som krävde mycket tid, främst spelaren ska tillhöra världen men bete sig på ett väldigt avvikande sätt.
Främst handlade det om att jag inte hade erfarenhet av att hantera en fysikmotor, under denna tid var det mycket i projektet som tog betydligt längre tid än vad jag hade tänkt. Jag hade svårt att förutse de problem som uppstod.

Dokumentation

Jag har känt att dokumentationen stjälpte mer än den hjälpte i större delen av projektet, jag hade svårt att hitta en balans mellan dokumentation och utveckling.
Utvecklingen blev många gånger lidande, jag var tvungen att först dokumentera att man ska göra något istället för att bara göra det direkt.
Däremot fick jag betydligt bättre balans i dokumentationen mot slutet av projektet.

Testdelen

Testningen är något jag kunde gjort mycket bättre och lagt betydligt mer tid på. Jag hade svårt att hitta testfall och arbeta utifrån ett testdrivet tänk.
Tror mycket att detta beror på att det varit så mycket nytt. Jag vet inte exakt vad som är bra kodstruktur i spel, och med spelbiblioteket och fysikmotor vet jag ännu mindre om bra kodstruktur .

Det är inte förrän sprint 7 och 8 som testning egentligen har varit aktuell, dels är det inte förrän nu som projektet har fått en så pass stor storlek att det kan vara vettigt att skriva automatiserade tester. Dessutom har inte projektet alltid haft den funktionalitet som, jag anser, bör testas. Skriva testfall på ett "Hello world" program känns ganska menlöst att testa.

Tidigare har jag känt att jag haft god översikt över de funktioner jag har implementerat tidigare testning hade mest resulterat i testning av inte implementerade saker, och testning av saker som inte är färdigutvecklade. Detta beror främst på att jag har svårt att få till en test driven utveckling, och ett testdrivet tänk.
Jag har i mitt projekt mer använt testning till att finna buggar.

Jag har implementerat en test-state där jag har gjort det mesta av min utveckling och testat den funktionalitet jag utvecklat, tanken är att jag ska använda denna state som testbädd senare om jag ska implementera mer funktionalitet

Sammanfattning

Övergripande är jag nöjd med mitt projekt och dess struktur. Det finns naturligtvis mycket delar som måste ses över för att dom ska fungera effektivare och bättre. tex en del klasser känns ogenomtänkta och hafsiga.
Jag tycker att jag har lagt en bra grund för att ta projektet vidare, det är ganska enkelt att skapa nya banor och lägga till mer funktionalitet

Hade i början tankar på att också göra en webbsida till spelet, men jag fick aldrig tid till detta. Jag valde istället att först fokusera på mitt spel och dess funktionalitet

Det har varit väldigt mycket som har varit nytt för mig och jag har saknat kunskap om hur jag gör vissa saker. Hade jag haft den kunskap jag har nu i början av mitt projekt hade jag naturligtvis lagt upp utvecklingen annorlunda och framförallt bättre på en gång istället för att implementera och använda klasser på ett helt värdelöst sätt och vara tvungen att göra om dem.

För att göra projektet bättre handlar det mest om att utöka mina kunskaper. Jag vet väldigt lite om spelprogrammering och hur man gör ett optimerat spel.

Jag har däremot blivit en bättre programmerare med hjälp av detta projekt och det har stundtals varit väldigt roligt.