

Dinosaur game

Mattias | 200s| 2021-05-23

Innehåll

[Inledning 2](#_Toc72955106)

[Projektbeskrivning 3](#_Toc72955107)

[2.1 Planering 3](#_Toc72955108)

[2.2 Uppbygggnad 3](#_Toc72955109)

[2.3 Koden 3](#_Toc72955110)

[Resultat 5](#_Toc72955111)

[Problem 5](#_Toc72955112)

[Avslutning 6](#_Toc72955113)

# Inledning

Under detta projekt har jag gjort ett ”runner spel” som går ut på att man styr en dinosaurie som ska försöka hoppa över kaktusar och ducka under fåglar för att komma så långt som möjligt. Mitt spel är väldigt inspirerat av Googles egna ”T-Rex game” och därför så kallas spelet för Dinosaur game. Alla sprites jag använt är egengjorda kopior av dem som är i originalspelet och jag har gjort dem genom att rita ut varje pixel med hjälp av bildredigeringsprogrammet Paint.NET. Hur dinosaurien styrs har jag även delvis tagit inspiration från originalspelet då dinosaurien som sagt kan ducka och hoppa, fast jag har valt att ändra på tangenterna så att man hoppar med W och duckar med S, då jag anser att det passar bättre för de flesta 2D-spelen har tangenterna för de rörelserna.

# Projektbeskrivning

Figur 1 - Planner

## 2.1 Planering

För att planera strukturera mitt arbete och bryta ner det i mindre delar så använde jag mig av Planner som är ett inbyggt planeringsverktyg som finns i Microsoft Teams. I Planner så strukturerade jag mitt arbete genom att dela in arbetet i 4 stora arbetsområden som behövde göras för att spelet skulle nå vad jag ansåg var spelets minimumgräns. Dessutom gjorde jag en bonuslista med saker jag skulle göra ifall jag hade tid över (se Figur-1). Inom dessa arbetsområden delade jag sedan upp arbetet ytterligare och la in mer ko0nkreta underrubriker så att jag visste vad jag skulle arbeta med.

## 2.2 Uppbygggnad

Spelet som är ett 2Dspel är gjord med hjälp av Unity som är en spelmotor man kan utveckla både 3D och 2Dspel med. Jag använder endast en scene, vilket är vad man ser när man spelar spelet och alla objekt jag använder är egenritade sprites som jag gjorde med hjälp av Paint.NET. Eftersom jag bara använde en scene och ville ha en gameovermeny så har jag gjort så att ”gameover texten” och ”restartknappen” ligger som childs i en ”gameroverscene” som är osynlig fram till att dinosaurien träffar något hinder och då blir den synlig tills man startar om spelet genom att klicka på omstartsknappen. De enda permanenta objekten under hela spelets gång är dinosaurien som aldrig rör sig annat än upp och ner och molnen som ständigt rör sig åt vänster och sen teleporteras tillbaka till höger om vad man ser så att de ständigt åker moln åt vänster för att ge en effekt av att dinosaurien springer. De andra objekten, dvs hindren, består av två olika sorters kaktusar och en fågel som alla är prefabs som spawnas med hjälp av en osynlig spawner.

## 2.3 Koden

All kod som styr vad alla olika objekt i Unity gör är skriven i Visual Studio Code som är en programmeringsutvecklingsmiljö där man kan programmera alla möjliga saker. Koden som styr de olika objekten är indelade i olika scripts som de olika objekten har. Spelet består av 5 olika scripts, dinoscript, birdcactus, cloudscript, spawnerscript och scorescript. Som man hör på namnet styr dinoscript dinosaurien och det är den som avgör hur dinosaurien rör på sig fast det är även det scriptet som avgör när dinosaurien träffar ett objekt och då startar funktionen ”Dinohit” som fryser spelet och startar ”gameoverscenen”.

Birdcactus och cloudscriptet styr fåglar och kaktusar respektive molnen och deras enda funktion är egentligen att röra objekten åt vänster. Den enda skillnaden mellan dem är att birdcactus raderar objekten efter att man inte ser dem längre för att förebygga lagg medan cloudscriptet teleporterar molnen till höger om vad man ser så att de hela tiden rör på sig.

Spawnerscriptet är scriptet som styr hur kaktusarna och fåglarna spawnas, dvs skapas. I det scriptet så finns det varje 1,5 sekund 75% chans att det spawnas ett nytt hinder. Hindret som spawnas är helt slumpmässigt valt från en array(en lista) med olika prefabs som jag lagt in i unity och koden är skriven på ett sådant sätt att man inte behöver uppdatera den när man lägger in fler objekt vilket jag gjort för att underlätta ifall jag hade velat lägga till något mer hinder.

Scorescriptet styr två stycken ”textrutor” som man ser uppe till höger i spelet och den längst till höger visar scoret man har under detta försök medan den vänstra visar ens egna highscore. Scorescriptet uppdaterar ens score medan man springer och var hundrade poäng så spelas det ett ljud för att indikera att man fått hundra poäng. Highscoret uppdateras så fort man förlorar och har mer poäng än vad man hade innan och Unity lagrar ens highscore med hjälp av playerprefs.

# Resultat

Resultatet vart väldigt lyckat då jag lyckades åstadkomma allting jag ville med spelet. Till en början hade jag planerat min minimumgräns och några bonussaker jag ville implementera, och jag nådde inte bara med minimumgränsen utan lyckades implementera alla bonussaker jag ville också som t.ex Highscoreräknaren. Hade jag gjort detta projekt igen tror jag att jag egentligen inte hade gjort själva spelsättet på något alltför annorlunda sätt och istället för att göra som nu och mer eller mindre sno alla sprites och ljud från Google Dino Game hade jag velat ge alla sprites och ljud något annat tema så att det mer hade blivit som mitt egna spel istället för en kopia av ett redan existerande spel.

# Problem

Under detta projekt har jag stött på många olika problem både i Unity men även utanför Unity. Eftersom jag tog mycket inspiration från Google Dino Game och ville använda deras sprites så ritade jag alla bilder i Paint.NET vilket var väldigt svårt och tidskrävande och för att lösa det problemet så fortsatte jag egentligen bara att rita om och om igen tills det vart på sättet jag ville. Nu i efterhand har jag dock insett att alla dessa sprites redan fanns tillgängliga i ett ”spritesheet” och om jag gör ett annat spel så tänker jag nog försöka hitta redan existerande sprites så att jag kan lägga mer tid på programmeringen.

Eftersom detta var mitt första projekt i Unity stötte jag på ganska många problem där som jag ganska snabbt överkom, fast det var ett problem som tog väldigt lång tid vilket var animeringar. Det jag inte lyckades med var till en början att ens få det att funka över huvudtaget då jag inte själv kunde förstå hur animeringspanelen fungerade och hur animationerna var uppbyggda. För att lösa det så följde jag en youtube tutorial av Brackeys[[1]](#footnote-2). Fast när jag väl lyckats få animationerna att ske så stötte jag på ett annat stort problem med animationerna, hitboxen uppdaterades inte när jag duckade! För att lösa det problemet gav jag min dinosaurie en box collider, så att den sammanlagt hade två colliders, en box och en polygon. Efter det så valde jag att box collidern som motsvarade hitboxen dinosaurien skulle ha när den duckade endast skulle vara på medan man duckade och då stängdes den vanliga polygon collidern av.

Själv kodningen var även till en början klurigt då jag inte hade kodat med C# innan, även om jag kodat lite i C++ som liknar C#, vilket ledde till att jag hade lite problem med syntaxen fast efter ett tag vande jag mig. Det jag hade mer problem med var att programmera alla scripts då jag aldrig hade hållit på med sådant innan och därför består min kod till största del av grundkod jag hittat online som jag sedan modifierat för att passa mitt spelprojekt.

# Avslutning

Till en början trodde jag inte att resultatet skulle bli så lyckat som det blev då jag nästan vart överväldigad av Unity fast nu efter detta projekt har jag både lyckats framställa ett eget spel och lärt mig mycket nytt, både i Unity, Paint.NET och C#. Hade jag gjort om detta projekt hade jag som sagt använt andra sprites så att spelet hade känts ännu mer som mitt eget fast även om spelet kan kännas som en kopia är jag väldigt nöjd med det slutgiltiga resultatet! Planeringen i Teams var nog ändock det som gjorde att det hela fick en bra struktur, jag hade genom den en övergripande idé om vad som skulle genomföras och uppdelningen i små delmoment gjorde att när jag arbetade med ett moment hade jag en klar bild av vad momentet skulle göra och dess angränsningar till övriga delmoment/programdelar.

1. <https://www.youtube.com/watch?v=hkaysu1Z-N8> [↑](#footnote-ref-2)