DonutRain

DonutRain er et spill som går ut på å samle poeng og liv ved å fange donuts som daler ned fra himmelen og fanges i en bøtte, samt å unngå å fange brokkoli som gjør at spilleren mister liv.

Prosessen

Eksamensgruppen ble dannet nokså tidlig i semesteret, vi møtte opp jevnlig etter forelesninger, gjerne i Mathallen der vi kunne belønne oss selv med litt donuts en fredag ettermiddag. Og i idemyldringens spede begynnelse fant vi ut at uansett hvordan spillet skulle ende opp, så skulle donuts være et hovedtema.

Vi hadde flere fysiske møter for å legge en plan på hvordan eksamen skulle utføres. Da Corona-situasjonen oppstod mistet vi litt motivasjonen fordi vi så for oss at det kom til å bli vanskelig å jobbe så tett uten å kunne møtes, men vi oppdaget raskt at Zoom-møter, Facebook-chat og en Unity-gruppe på Canvas fungerte veldig bra for oss. Etter mye idèmyldring landet vi på et klassisk «catch falling items»-spill. Visjonen og stikkordene vi jobbet ut fra var et retro game-boy plattformspill, blandet med et rosa candy-land.

Vi fordelte ansvar etter egne ønsker og etter hvor stødige vi følte oss på koding/unity. Selv om noen hadde hovedansvar for scripting, ble store deler av kodingen gjort i felleskap, slik at alle hadde en forståelse for hva vi gjorde og hvordan ting fungerte. Dette gjorde vi ved å dele skjerm i Zoom-møter, så alle kunne komme med innspill og det ikke ble så mye deling frem og tilbake på selve prosjektmappen. Ettersom vi ikke satte oss inn Unity-collab, heller ikke GitHub, ble dette en løsning som fungerte veldig bra for vår gruppe. Vi lastet opp mappen i en gruppe på Canvas etter store endringer var gjort, og ga de ulike versjonsnavn etterhvert som vi jobbet. Vi delte også andre assets vi ville bruke i spillet her. Videre satte noen seg mer inn i spesifikke Unity-elementer, som bl.a particle-systems, lyd, UI osv.

Utvikling av spillet

Vi startet dette prosjektet med å se for oss en begrenset tidsperiode spilleren ville ha på seg for å fange flest donuts, men gikk etterhvert bort ifra det. Vi ønsket i stedet at man skulle få poeng for hvert objekt som landet oppi en bøtte. Vi ville også ha med et objekt som skulle ta bort liv dersom det ble fanget, for å gi spillet mer mening, og et objekt som både ga poeng og ekstra liv. Vi valgte donuts-objekter for å gi poeng, en gull-donut som ville gi ekstra liv, og et brokkoli-objekt for å ta bort liv.

Etter vi hadde kartlagt hva spillet skulle handle om var neste steg å finne de objektene vi ville at spilleren skulle fange/catche, og noe å fange de oppi. Hele gruppen jobbet med å finne noe som passet inn i med det estetiske uttrykket vi ønsket i spillet, eller som vi kunne gjøre om senere med materialer vi laget selv. Målet var at alle assets vi lastet ned skulle bli litt «vårt eget» selv om vi ikke laget selve modellene/assets selv.

Vi begynte så med å plassere en bøtte på et underlag, lage et GameObject/"spawning-objekt" som objektene skulle falle fra, og et form for poeng og liv-system. Når vårt spawning-objekt var plassert riktig ut i fra bøtten og produserte fallende donuts og brokkoli, måtte vi få de til å bli registrert når de havnet oppi selve bøtten, for så å forsvinne etterpå. Vi laget en collider med en tilhørende tag, som vi hentet i scriptet. Da kunne vi bruke denne til å hente ut info om objektene som «krasjet» i den, legge til betingelser for eventuelle poeng, minus liv osv. Vi laget så et script til selve bøtten, som vi også hentet med tag, slik at vi kunne styre den med tastaturet. Senere fant vi også ut at vi trengte en collider som ødela de items som ikke landet på vår collider i bøtten. Dette for å forhindre at for mange kloner av prefabs samlet seg opp og potensielt krasjet en uheldig spillers pc hvis spillet stod på for lenge.

Vi la til grenser (LeftLimit, RightLimit) i bildet som stod i forhold til lengden av spawner, som vi koblet opp til bøtten for å begrense området man kunne styre bøtten (forhindre at den forsvant ut av bildet). Første metoden vi brukte gikk ut på å lage et lite område på hver side av bildet som skulle trigges når bøtten traff det og dermed stoppe den, men vi opplevde at noen ganger fungerte grensene og andre ganger ikke. Først trodde vi at dette var en glitch i Unity, men vi fant til slutt ut at grunnen var "lagging". Hvis spillet "lagget" akkurat når bøtten traff grensen, så rakk den ikke å registrere at bøtten passerte grensene.

Neste punkt på listen var å koble sammen poengscore og liv til en UI (User Interface), slik at spiller kan se hvor mange poeng/liv h*n til en hver tid har. Vi ville også ha en form for hovedmeny og pausemeny. Vi ville at hovedmenyen skulle inneholde en info-oversikt over

hvordan man spiller og hva spillet går ut på, samt en oversikt over hvor mange poeng/liv de forskjellige items gir. Vi ville også ha en quit-knapp her, som avsluttet hele applikasjonen. Pausemenyen skulle trigges av «space»-knapp og kunne starte spillet igjen på samme måte, i tillegg til å trykke på den fysiske tilbakeknappen. Videre ville vi at spiller kunne justere lyd, og fant det mest naturlig at pausemenyens quit-knapp sendte deg tilbake til hovedmenyen.

Nå som spillets hovedfunksjoner var på plass, kunne vi begynne å utforme bakgrunnen/verdenen. Vi vurderte og testet mange ulike elementer for å ha i bakgrunnen, trær/skog osv. Besluttet å utnytte de donut-assets vi allerede hadde lastet ned, for å gi de flere bruksområder men også fordi det passet perfekt for temaet. Vi fant en fargerik sky-box som gav oss fargene vi ønsket, samt lys. Vi laget også en shader, som vi brukte for å få litt endring i bakgrunn fra hovedmeny til spill-scenen.

Lyd

Ettersom lyden i et spill er med på å sette en atmosfære og ikke minst kan gi betydningsfull stimuli, ble det viktig for oss å finne lyder som ville stå i stil med resten av spillet. Det var viktig at lydene ville passe til spillet på både det estetiske planet og ut ifra den opplevelsen vi ønsket at brukerne skulle få ved å spille spillet.

Vi valgte en bakgrunnslyd som ga en «old school / gameboy» følelse, noe vi tenkte ville passe godt til spillets utseende og karakter. Bakgrunnsmusikken skulle spille kontinuerlig gjennom hele spillet (loop), samt spilles av med en gang spillet ble startet opp (Play on awake).

Videre la vi også til lyd på alle knapper i de ulike menyene, start-meny, pause-meny og gameover-meny. I pause-menyen la vi i tillegg til en slider hvor spilleren selv kan justere lydnivået på bakgrunnsmusikken. Til dette brukte vi UI-slider og music mixer i Unity. Ettersom music mixer i Unity er basert på en logaritmisk skala, mens slideren er basert på en lineær skala la vi til et “SetVolume” script som konverterte mixeren til å gå ut i fra samme verdi som slideren. Dette gjorde at slideren ble mindre sensitiv og at volumet justerte seg mer jevnlig.

I tillegg til bakgrunnsmusikk og lyder på menyknapper ønsket vi å legge til lyder på alle fallende items som skulle spilles av med en gang de traff bøtten i spillet. Lydene vi valgte til

items var i samme sjanger som bakgrunnsmusikken og skulle signalisere for spilleren om det var positivt eller negativt å fange et item. Vi ga de vanlige donutsene en positiv lyd, mens brokkoli fikk en lyd med en litt mer negativ karakter. Gull-donuten ga vi også en lyd ved spawning for å understreke at det kom et item det var en fordel å fange, samt en lyd når den traff bøtten som skulle indikere en økning i liv.

Particle system

Ettersom gull-donuten ville være ekstra ønskelig for en spiller å fange i spillet, ville vi forsøke å fremheve denne litt mer enn de andre items, på en måte som ville gjøre det tydelig for spilleren at den var spesiell. Vi brukte derfor particle systems for å legge til en effekt som gjorde at gull-donuten fikk «glitrende» partikler rundt seg mens den falt nedover. Vi gjorde partiklene relativt små og laget et materiale som ga en glødende effekt. Vi satte «levetiden» på partiklene til 2, da det ikke var behov for mer, og satte «start lifetime», «startspeed» og «start size» mellom to variabler for å lage en mer tilfeldig partikkeleffekt.

Det ble en del prøving og feiling med particle system når det kom til innstillingene i Inspector View, både på «rate over time», hvilken form vi ønsket at partiklene skulle skytes ut i og hvordan vi fikk lagt til det materialet vi ønsket. Resultatet ble kanskje ikke helt slik vi hadde sett for oss, da vi ønsket en mer glødende/sky-aktig effekt, men alt i alt er vi fornøyde med hvordan det ble. Partikkeleffekten gir i det minste et tegn til spilleren om at gull-donuten er litt mer spesiell enn de andre donutsene og verdt å fange.

Et at de siste stegene i utviklingen av spillet var å sette inn en game-over funksjon som stoppet spillet, når spilleren var tom for liv. En game-over meny dukker da opp og spilleren får valg om å prøve på ny, eller gå tilbake til hovedmeny.

Utfordringer

En av hovedutfordringene vi støtte på var å kunne dele opp den såkalte «spawn-raten» altså hvor ofte et element skulle dukke opp. Vi ønsket at ting skulle falle tilfeldig og hadde scriptet elementene deretter, men ønsket f.eks at gulldonuten skulle dukke opp en sjelden gang, og at brokkoli skulle dukke opp hurtigere for større vanskelighetsgrad. Vi slet veldig med hvordan

dette skulle gjøres. Det beste vi kom fram til var å legge 3x spawnItems script på spawneren, som da dannet 3 ulike lister, med en tilhørende variabel for spawnrate på hver av disse. Dette fører til at elementene spawner i forhold til raten i sin liste og ikke i forhold til hverandre. Det gjør også at det starter 3 ulike prosesser hver gang man starter spillet, så det dukker alltid opp en vanlig donut, en brokkoli og en gull-donut. Dette er en «bug» vi er klar over, men den beste løsningen vi klarte å få til.

Det andre hovedproblem/utfordring kom når vi skulle få til flere levels/level-up. Vi hadde sett for oss lenge at det «bare» var til å lage ny scene, med samme oppsett som skulle loades når man nådde en viss poengsum. Deretter endre spesifikke innstillinger her som speed på items, spawning osv. Det vi ikke kom på var at ny scene da også resetter variablene som holder på poeng/liv. Vi prøvde på en del forskjellig for å ta vare på variabler mellom scener, men som også måtte kunne resettes når man startet hoved-spill scenen på ny, eller fra hovedmeny. Eksperimenterte bl.a med playerrefs, som kunne vært interessante for eventuell videre utvikling av spillet,for å kunne ha high-score oversikt, ulike saver/kunne slutte spillet og fortsette. Men til slutt innså vi at tid og ressurser ikke strakk til, og vi beveget oss litt utenfor pensum og vårt kompetansenivå. Beste løsning ble da en level-up funksjon i samme scene. Vi skulle likt at det var kun en funksjon «LevelUp» og ikke to ulike, som justerte seg opp i det uendelige per f.eks 20. poeng. I siste liten rakk vi å oppdage at pause-menyens «resume» var satt til en time-scale = 1. Dette gjorde da at man endret hastigheten tilbake selv om man hadde gått opp i level.

Vi hadde også en liten utfordring med å kontrollere at donuts beveget seg i den hastigheten og retningen vi ønsket. Vi slet også med å kunne få donuts/items til å rotere (at de ikke kun falt rett ned uten å bevege seg), og heller ikke lande utenfor området til bøtten (på bakken). Til slutt fant vi ut at retningen de faller måtte være i forhold til verdenen og ikke seg selv. Med fysikkens lover i bakhodet roterer ikke en brokkoli på samme måte som en donut ville gjort i virkeligheten, slik som vi har kodet at den skal i spillet. Fordi den har et annet tyngdepunkt og den roterer rundt en ulik akse enn de andre(donuts). Men dette velger vi å overse i vårt Donut-land, ettersom fysikkens lover her er litt annerledes.

Til tross for mye frustrasjon rundt disse utfordringene, er kanskje det at vi klarte å finne en løsning på disse noe vi er mest stolte over. Vi er også veldig fornøyd med utformingen av "bruksanvisningen" da den er grafisk tilfredsstillende samt veldig enkel å forstå og gir

spilleren en god innføring i hvordan spillet fungerer. Sluttresultatet samsvarer i stor grad med det vi så for oss før vi startet og vi føler det reflekterer en god planlegging i bunn.

Liste over brukte unity-elementer i prosjektet

- 3D elementer: modeller, prefabs
- Game Objects
- Anvendt teksturer og laget materialer, shader.
- Sprites
- Scene management
- Rotation, transition og scale av objekter.
- Particle-systems
- Audio
- Collider
- Tags
- UI-elementer: input fra bruker, canvas, panel, buttons, text, image.
- Time.scale
- Slider /Music Mixer

Kildehenvisning

Skybox
https://assetstore.unity.com/packages/2d/textures-materials/sky/skybox-add-on-136594

Donuts
https://www.turbosquid.com/FullPreview/Index.cfm/ID/656913

Brokkoli
https://assetstore.unity.com/packages/3d/props/food/food-grocery-items-low-poly-75494

Underlag

https://assetstore.unity.com/packages/3d/environments/landscapes/low-poly-simple-nature-pack-162153?fbclid=IwAR1FiPzhjI-NR-DfaLv6xvXH_XnCZP-NdyH6Evk2YRoxXKzkwNfwm2kBJ2M

Basket/bøtte

https://assetstore.unity.com/packages/3d/props/cartoon-props-pack-18677?fbclid=IwAR22eBj8kuRdTwRCSgctWBlWK_8icBNfmC0PgUjLAtiEqruhSAm9MtEFB0Q

Font

https://assetstore.unity.com/packages/2d/fonts/free-pixel-font-thaleah-140059?fbclid=IwAR1MO-SXtPE01_-wREePGnOdatza7BI1Cr4tTn0ctHX_x0x3ZoJGMyN-H-E

Lyd

https://www.zapsplat.com