Rapport

s181086 Svein Gunnar Fagerheim
s181087 Alf-André Walla

Beskrivelse av prosjektet
Vi har laget en start på et spill som midlertidig kalles Super Mario Ultra, og er en mario-basert plattformer, bare med uendelig randomisert terreng.
Dette er ganske likt den klassiske Super Mario. Vår versjon har ikke noe plott ennå, det handler for det meste å gå lengst mulig til høyre på bildet. Man gjør dette med å gå og hoppe alt etter hvilke hindringer man møter på. Mario Har også forskjellige powerups, med sopp blir har stor, med blomst kan han skyte ildkuler og stjerne gjør han uovervinnelig en stund. Han kan også miste powerups ved å bli skadet, med unntak av stjerne (som gir midlertidig udødelighet).

Kompilering
Prosjektet er blitt kompilert for linux, men pga. vanskeligheter med drivere har man ikke fått kjørt det der.
Prosjektet er blitt testet på windows, og kjører fint der, hvis man har en GPU som støtter minst OpenGL 3.x (eller senere).

Linux: Installer x11, xrandr, glfw, og drivere til skjermkort. Installer så BASS:
Installering av BASS kan gjøres manuelt eller ved å kjøre bassinstall.sh fra /lib
Ubuntu package til glfw heter muligens glfw-dev
Hvis alt er på plass er det bare å kjøre ‘make’
Hvis display-drivers er på plass skal programmet kjøre native linux. Hvis ikke får du fortsatt opp glfw-vinduet, men en feilmelding som sier at driverne ikke ikke støtter extension gpu_shaderx. (Eller noe lignende.)
Vi har fått det til å kjøre på linux uten problemer. Fant ut at getline() ikke fjerner \r\n, men kun \n, og brukte en hjelpefunksjon som bytter ut getline() med getSafeline(). Dermed var det kompilert og kjørt på både Windows og Linux!

Grunnen til vanskelighetene med kompilering på linux er avhengigheten til X11 for vindu og div human interfaces.

Windows: Kjør build.bat, start game.exe

Eksterne biblioteker
BASS er det eksterne biblioteket vi bruker for lyd (musikk, soundclips), og er multiplatform. BASS er bass.dll / bass.lib på windows og libbass.so på linux. (Derfor må man rekompilere ved oppdatering av libbass på linux, men kanskje ikke på windows, kommer an på .lib)

GLFW er et lightweight multiplatform OpenGL context program som manager ett vindu, leser keyboard/joystick og kan hente ut GL extensions (som returneres som function pointers). OpenGL er et C-library (og vil alltid være det,) så man må kompilere med -Wno-write-strings for å få lov til å bruke C-strenger direkte inn i funksjonskall.

Feil og mangler
Fysikken i spillet er feilkonstruert, på grunn av tidsmangel. Akselerasjon ble switchet ut med konstant hastighet for å simplifisere problemene. Alpha release! All akselerasjons programmering er på plass, det må bare tunes hardt.
Negative posisjonsverdier (x-akse) har en floor() feil, dvs hvis spiller går bakover lenge vil verden tegnes på rett plass, men player vil testes mot (int)float istedet for floor(float), og vi er i innleveringsmodus. Eks.: (int)-1.9 = -1, floor(-1.9) = -2, (int)2.9 = 2, floor(2.9) = 2, der rette svar er -2 og 2.

Uferdige (mangelfulle) features
Alt er på plass for å få inn flere spillere, men vi fikk aldri tid til å benytte config variablene og lage en egen kontroller klasse der man kunne velge keyboard / joystick.
Mye er på plass for å få inn sopper og penger, men mekanikken må lages og rendring av ‘units’. (Animerte)
Fiender med litt AI og mekanikk er også units, men er mer arbeid, og ville tatt mye arbeid.
Mye tegnearbeid gjenstår på spilleren for å få inn blant annet løping og “stor mario.” Stor mario er i stor grad ferdig, og er blitt implementert. Fokuset ville uansett blitt på å få inn penger, poenger og sopper.
Fonter er mye arbeid å få inn i en rasterizer, og ville havnet langt nede på lista.

Ting som finnes bedre løsninger på
Render-bare objekter har ikke en felles klasse som de kan arve, dette fordi alle _forskjellige_ typer objekter har forskjellige måter å rendres på. Med litt (mye) mer tid kunne det kanskje blitt en komplisert løsning på dette, men det var nok av arbeid fra før av.

Siden det har gått mye tid på å lage “skjelettet” til en platformer, er det mange løsninger som kunne ha vært bedre, bl.a. ble det ikke tid til å gjøre om matriser og vektorer til templates. Koden hadde i tillegg blitt litt mer slitsom da det egentlig bare var bruk for float for begge.

Generatoren som lager verdenen kunne ha vært mye mer avansert, bla.a. med tanke på overganger fra grønt til vinter osv. Dette hadde kommet til å ta tid, men er fullt mulig å gjennomføre.

TTT - ting tar tid.

Coding practices

- Namespaces, klasser, funksjoner og variabler skal ha deskriptive selv-forklarende navn.
- Enkapsulering skal bevares, og all mekanikk skal isoleres fra rendring. Det siste på grunn av at man vanligvis kjører 2 tråder, en for rendring og en for alt annet.
- Kommentarer kun for å forklare hvorfor, eller potensielle problemer, påminnelser og slikt.
- Hver modul må testes, og man gjør ferdig det man begynner på. Dette kan ses iom. at alt som er inne - fungerer.
- Inkluder alle headere som brukes uansett om man headeren er inkludert av en annen. (Unntak for glfw, som alltid inkluderer GL/gl.h)
- Ikke inkluder headere der man kun trenger forward. (Dette sliter vi litt med, men det kommer vel etter hvert)
- Bruke kun nødvendige portable libraries, dvs kun lyd og opengl framework.