I Am not Legend

Af Jeppe Møller Wortmann

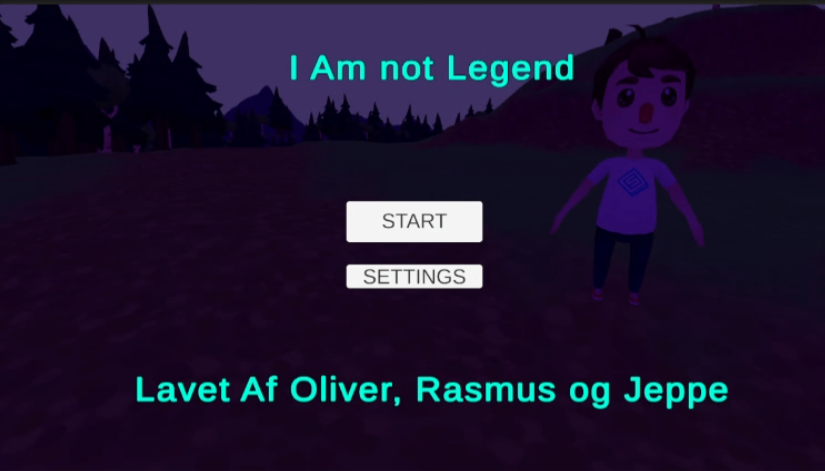
27/04/2023

Projekt lavet med

Oliver Mads Kragh

Rasmus Riemke Rubæk

Med fokus på Playercontrolleren, UI, og våben



## Projektbeskrivelse

Vi valgte netop at lave vores program fordi alle gruppemedlemmer interesserer sig for zombie-genren. Dette program inkorporerer også en stor del af de funktioner og lignende, vi har lært i de forskellige forløb.

Vi vil udarbejde et program, der minder om Call of Duty: Zombies, men med en anden målgruppe, og mindre avanceret. Målgruppen er mere fokuseret på personer der er nye til genren.

Vi vil redegøre for programmets udvikling, heri med fokus på selve koden og opsætningen.

Vi vil undersøge, hvorvidt oplevelsen med spillet kan forbedres ved hjælp af flere forskellige sværhedsgrader.

Vi vil diskutere hvorvidt produktet kan vække en interesse for zombie-genren.

Vi vil diskutere hvorvidt spillet kan udvides med flere funktioner.

Jeppe Møller Wortmann vil i sin synopsis besvare problemstillingen med fokus på Playercontrolleren, UI og våben.

Oliver Mads Kragh vil i sin synopsis besvare problemstillingen med fokus på Wavespawner og integrering af GitHub.

Rasmus Riemke Rubæk vil i sin synopsis besvare problemstillingen med fokus på troubleshooting, zombier, og 3D-objekterne.

## 

## Resume af projektet

Spiller et sat i en lille verden omringet af bjerge. I den lille verden er man ikke alene, som tiden går kommer en hær af zombier imod en, når man får dræbt alle zombier, kommer der en ny hær, denne gang med flere zombier. Spillet har UI både i form af en hovedmenu, hvor man kan ændre indstillinger, men også i form af informationer på skærmen når man er inde i spillet, de informationer er bl.a., hvor mange skud der er tilbage i magasinet, hvor meget liv man har og hvor mange zombier man har skudt.

I udviklingen af dette spil har jeg haft fokus på Playercontrolleren, våben og UI. Sammen med Rasmus Riemke Rubæk og Oliver Mads Kragh har vi hjulpet hinanden igennem udviklingen, og har på den måde fået lavet spillet.

## Playercontroller udvikling

Uden at kunne styre vores karakter, fungerer vores spil ikke. Til det har man Player Controlleren. Jeg startede med det basale, frem og tilbage. For at kunne flytte sig frem og tilbage skulle spillet først kende til vores input, derfor oprettes to variabler:

horizontalInput og verticalInput



Disse to variabler kan ikke bruges til noget hvis de ikke gives en værdi, det gøres i void Update for at variablens værdi altid svarer til den reelle værdi





For at undgå at skulle skrive 4 forskellige tilfælde af inputtet bruges Input.GetAxis, i stedet for Input.GetKeyDown. Hvis man havde brugt GetKeyDown skulle der være oprette 4 forskellige tilfælde af hvad spilleren skulle gøre, i stedet bruges GetAxis hvor 2 tilfælde er nok. som ses nedenfor:





De to inputs ændre fortegn hvis man trykker på den modsatte retning, derfor ændrer spillerens movement også retning

**Drej med musen**

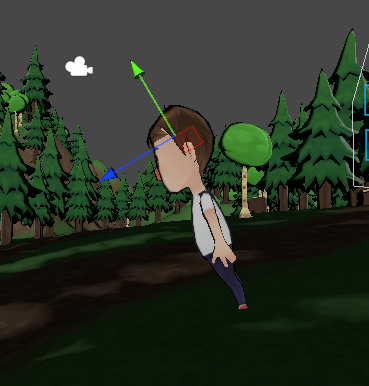
For at gøre spillet et rigtigt FPS-spil skal man kunne dreje spilleren med musen. For at få musens rotation, bruges igen GetAxis, denne gang findes musens akse X og Y akse i stedet





hvorefter transform.Rotate bruges til at rotere spilleren

At rotere spillerens x akse resulterede dog i at hele spillerens krop blev roteret, og det går ikke



For at løse dette problem blev kameraets og spillerens rotation gjort uafhængige af hinanden, drejningen i y aksen blev lagt ind i kameraet så det kun er kameraet der flytter sig, og ikke spillerens krop

**Camera**

Kameraet blev nu lagt ind under spilleren som et child, for at holde inspektoren fin



For at styre kameraet blev scriptet CameraController oprettet. Den vertikale rotation flyttes nu ind i kamera scriptet som gør at man kan kigge rundt uden at spilleren roteres.

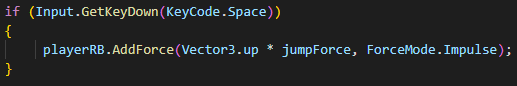
Endnu et problem blev fundet, ens mus rykkede rundt på skærmen, det resulterede i at man trykkede på forskellige ting på skærmen.

For at løse det bruges Cursor.lockState til at låse musen inde i spillet



**Movement**

Efter at have løbet lidt rundt på kortet føltes oplevelsen tom, det var for basalt. Derfor blev Hop og sprint tilføjet. Hoppet tilføjes med denne koden

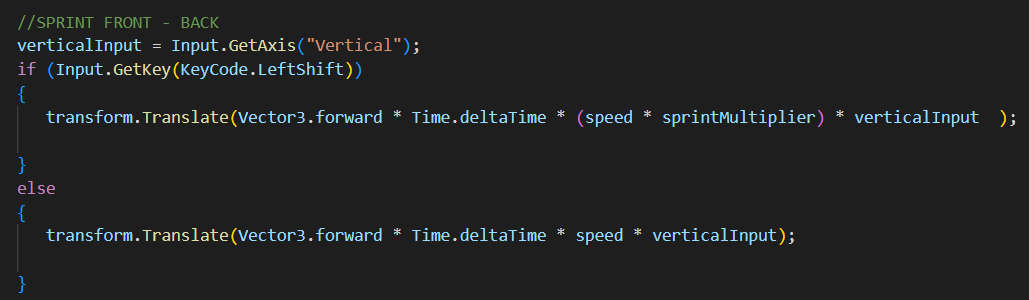


For at kunne tilføje en kraft til spilleren skulle man bruge en rigidbody, den hentes i void start således:

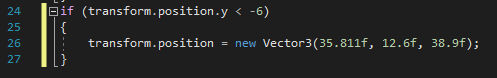


Variablen jumpForce blev tilføjet for at kunne bestemme hvor højt spilleren hopper.

**Sprint**

****

For at kunne sprintet blev der lavet et if - else statement, hvis shift bliver trykket ned, ganges speed med sprintMultiplier, for at gøre løbehastigheden større. Da vi testede sprint, opdagede vi at hastigheden var så stor at man kunne løbe igennem overfladen af map’et, det resulterede i at man faldt ned igennem mappet og blev ved med at falde ned for evigt. For at løse det blev løbehastigheden sat ned, det løste problemet. For at være sikker på at der er en løsning hvis problemet sker, blev der tilføjet noget kode der teleporterer spilleren op på mappet igen. Som en “undskyldning” til spilleren blev der valgt toppen af et træ som spawnpoint, der kan zombierne nemlig ikke komme op.



## 

## UI-udvikling

**Knapperne**

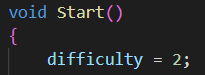
For at gøre vores spil nemt at interagere med, og forstå hvad der sker, skal der være et User Interface (UI). Til at starte med var min plan at lave en ny scene hvor startskærmen var. Efter at have opsat det viste det sig at være bøvlet at bringe informationer fra den ene scene til den anden. Derfor valgte jeg at lave en startmenu i samme scene som spillet.

Da spillet er målrettet en mindre avanceret gruppe, er startmenuen også derefter. Der er to muligheder, START eller INDSTILLINGER

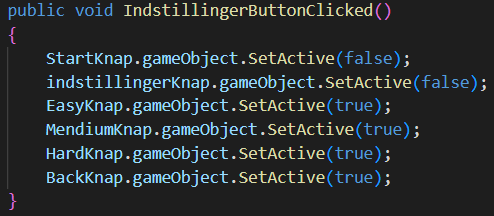
START starter spillet og fjerner menuen hvor INDSTILLINGER åbner indstillinger menuen, hvor man kan bestemme sværhedsgraden

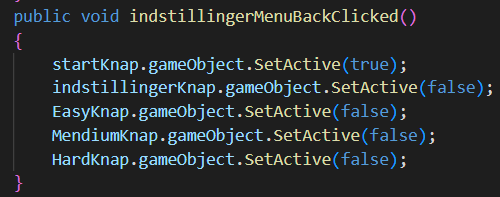
Hver knap sætter difficulty’en, easy er 1, medium er 2 og hard er 3.

Ved en test af spillet opdagede jeg at diffuculty’en kun blev sat hvis man trykkede på en af knapperne. Hvis man ikke valgte en difficulty var den 0. For at løse problemet blev difficulty’en sat i void start til at være medium.

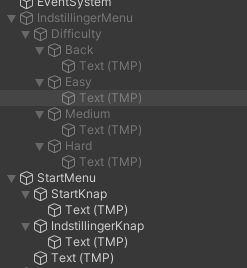


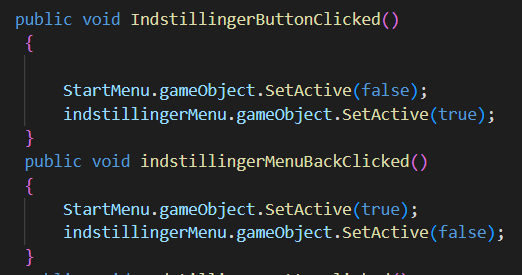
Når der skulle navigeres i de forskellige menuer, blev det gjort ved at sætte de forskellige knapper til at være inaktive eller aktive. I starten blev det gjort ved at ændre hvert objekt. (se figurer)





Da det var meget skrivearbejde indså jeg at man kunne lave et tomt objekt og sætte alle objekterne ind i et tomt objekt

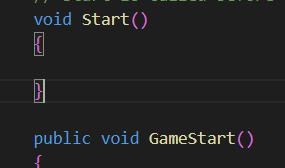


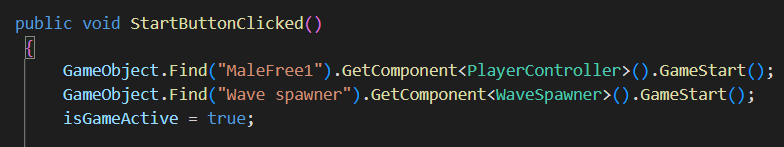


Det gjorde processen nemmere.

**Aktivt uden at trykke start**

Efter at teste startmenuen opdagede jeg at spillet stadig var aktivt selvom der ikke var trykket start. Alle controls til spilleren virkede stadig, og zombierne flyttede sig også. Derfor blev variablen isGameActive oprettet, sammen med voiden GameStart() i både playercontrolleren og Wavespawneren



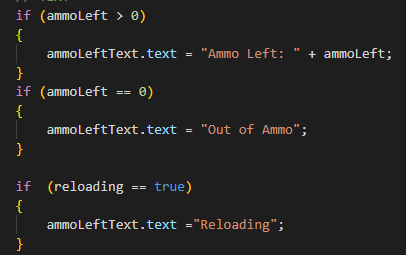


Det gør at de objekterne først starter når startknappen trykkes, derved blev problemet løst

**Skud i magasinet UI**

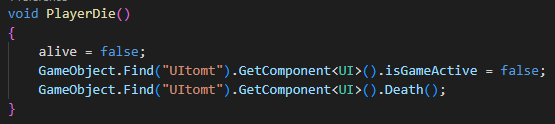
I denne del af udviklingen var jeg ved at have godt styr på UI-koden, derfor var der ikke længere en rigtig proces med fejl og løsninger.

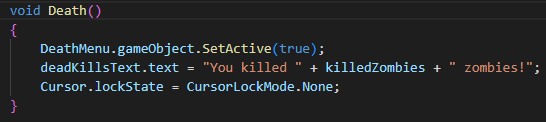
For at vise hvor mange skud man har tilbage i magasinet, bliver denne kode brugt:



**Dødskærn**

For at opsummere spillet når man dør er der lavet en dødskærm, den viser hvor mange kills du fik. Death() i UI startes at PlayerDie() i playercontrolleren

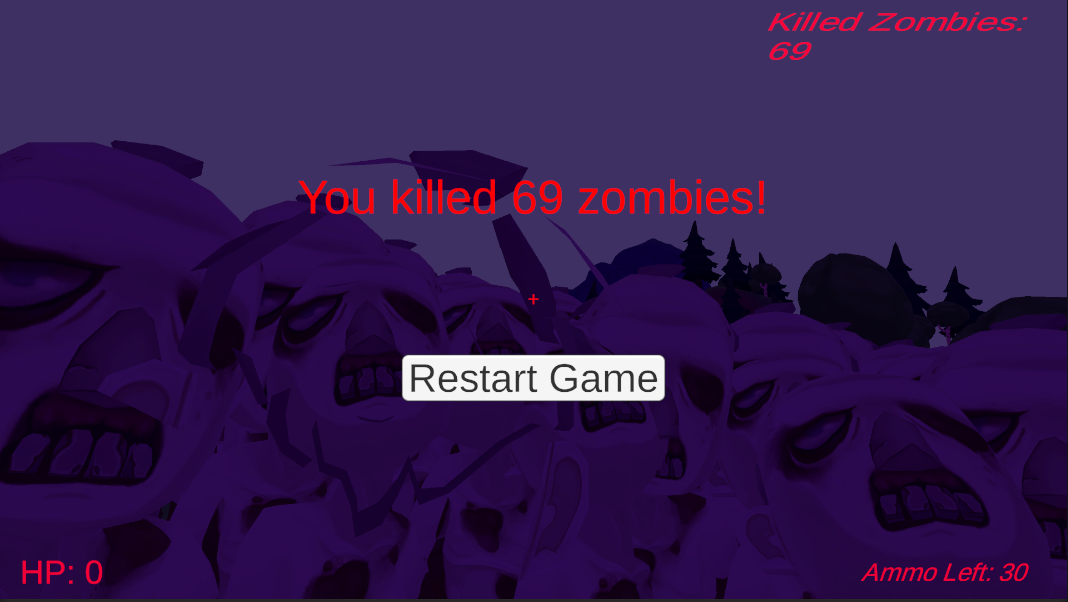




Dette er slutproduktet:



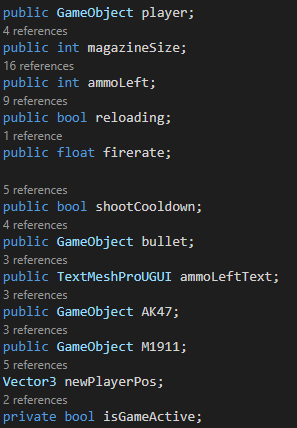




## 

## Deep dive af “Gun” script

### Variabler



player = spillerens figur

magazineSize = størrelsen på magasinet, den er public da det derved var nemmere at ændre magasinstørrelsen af de to våben. det er en int da det ikke kan være andet end et tal.

ammoLeft = mængden af ammonition der er tilbage i magasinet

reloading = en bool der beskriver om våbnet er ved at blive ladt, da det enten kan være ja eller nej er en bool valgt

firerate = hvor hurtigt våbnet skal skyde, den er også public for at kunne ændre nemt for hvert våben

shootCooldown = om våbnet skal vente med at skyde

bullet = prefabet af projektilen

ammoLeftText = det tekst objektet, dette skal findes da man derved kan vise i UI hvor mange skud der er tilbage i magasinet

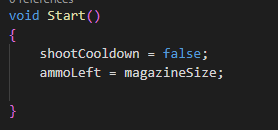
AK47 = Gameobjectet som er vores våben nr 1

M1911 = Gameobjectet som er vores våben nr 2

newPlayerPos = spillerens plasering med et offset

isGameActive = En bool der ved om spillet er aktivt, denne hentes fra UI scriptet

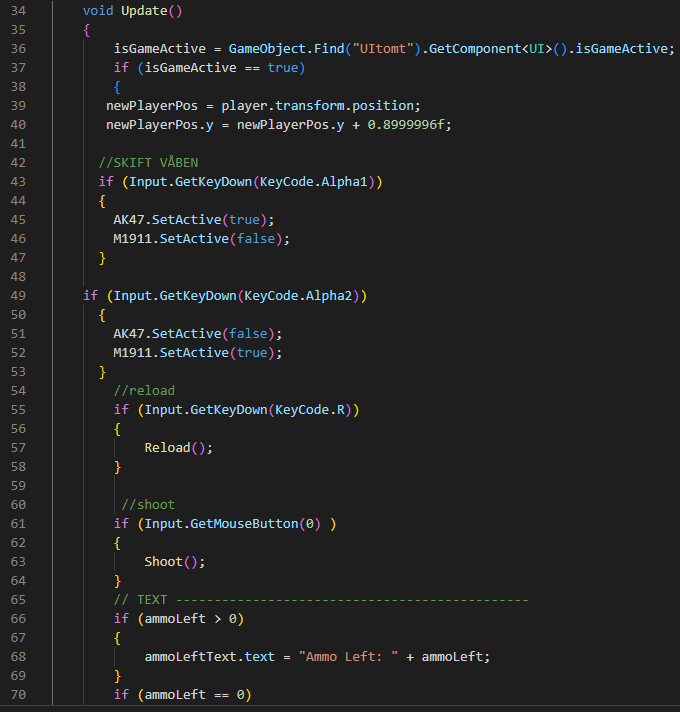
### void Start

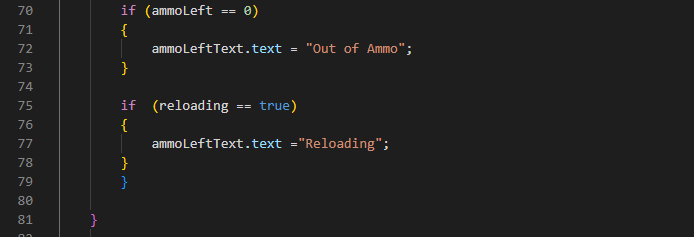


shootCooldown sættes i starten til at være false, da der jo ikke er skudt endnu

Mængden af ammunition der er tilbage sættes til at være magasinets størrelse

### void Update





Ting i void update bliver kørt hvert frame, i denne void er der derfor ting der skal opdateres tit.

36 isGameActive opdateres hver frame for at se om spillet er aktivt

37 hele update sættes til kun at køre hvis spillet er aktivt, det gøres da det var den nemmest måde at pause våbnet når man var i menuerne.

39 variablen newPlayerPos sættes til at være spillerens position

40 newPlayerpos.y sættes til at være en lille smule over, da det da passer med spillerens kamera. Det gøres da der senere i scriptet bliver den position til at spawne skuddet ved

43 Hvis man trykker på 1 vælger man AK47’eren som sit aktive våben. Med .setActive(true) sætter man våbnet til at være aktivt i Unity og dermed blive renderet.

44 Det modsatte gøres ved m1911’eren den sættes til at være false, altså uaktiv

49 Hvis man derimod trykker 2 gøres det modsatte af hvad der skete i 43 & 44

55 Der tjekkes om knappen R bliver trykket ned, hvis den gør, bliver Reload() startet

61 Der tjekkes om venstre museknap er trykket ned, ikke om den bliver trykket. Det gør at Shoot() er aktiv så længe musen er trykket ned

65-78 står for teksten på UI’et

66 er aktiv hvis der er mere end 0 skud tilbage i magasinet, hvis der er det opdateret ammoLeftText

68 text delen af objektet ammoLeftText sættes til at være “ammo Left:” og mængden af ammunition tilbage i magasinet

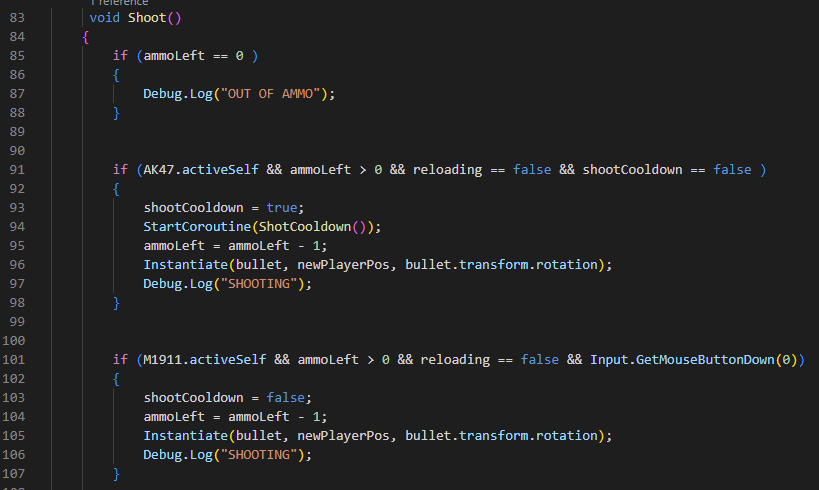
70 Denne del er aktiv hvis der er 0 skud tilbage i magasinet

72 tekst delen af objektet ammoLeftText sættes til at være “Out of Ammo”

75 Denne del er aktiv hvis våbnet lades

77 tekst delen af objektet ammoLeftText sættes til at være “Reloading”

### void Shoot



91 Denne linje styrer hvad der skal ske hvis både AK47 gameobjectet er aktiv, det gøres med .activeSelf. Da våbnet ikke kan kunne skyde uden ammunition i magasinet tjekkes der også for at der er mere end 0 ammunition i magasinet, da man ikke skal kunne skyde når der lades tjekkes der også at reloading == false. Hvis alle disse kriterer er oplyst kan våbnet skyde

93 Shootcooldown sættes til at være sand da der skal gå noget tid før der skydes igen

94 Coroutinen ShotCooldown() startes, i den routine ventes der

95 mængden af ammunition i magasinet gøres 1 mindre

96 Instantiate bruges til at lave skuddet, prefabben bullet bliver spawnet i koordinaterne newPlayerPos.

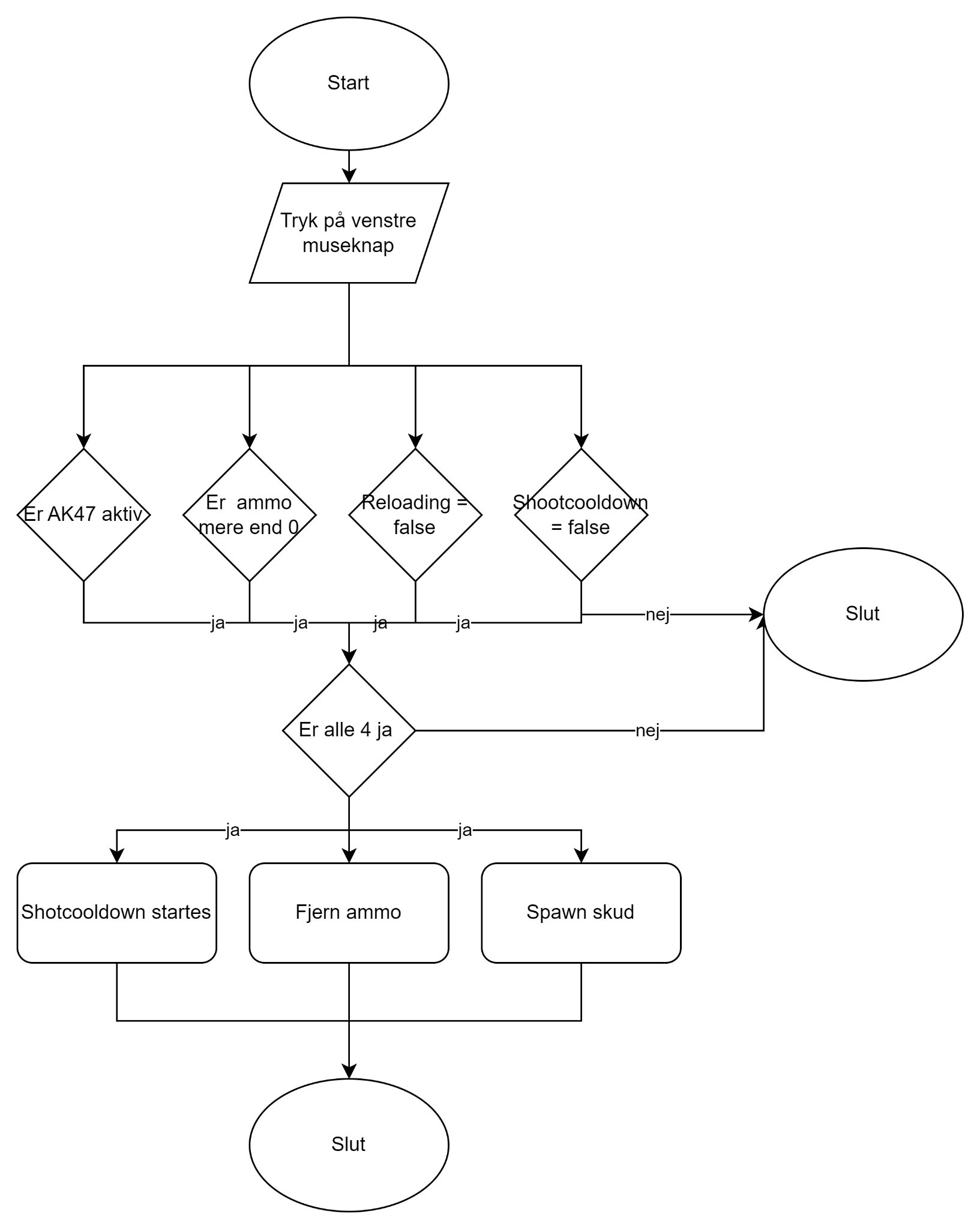
101 Denne linje styrer hvad der skal ske hvis både m1911 gameobjectet er aktiv, det gøres med .activeSelf. Da våbnet ikke kan kunne skyde uden ammunition i magasinet tjekkes der også for at der er mere end 0 ammunition i magasinet, da man ikke skal kunne skyde når der lades tjekkes der også at reloading == false. Da man ikke skal kunne skyde med pistolen ved at holde musen tjekkes der også for at musen bliver trykket ned.

103 shootCooldown sættes til at være false

104 mængden af ammunition i magasinet gøres 1 mindre

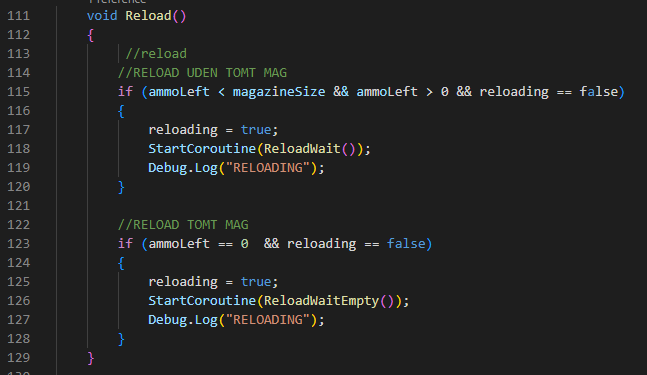
105 Instantiate bruges til at lave skuddet, prefabben bullet bliver spawnet i koordinaterne newPlayerPos.

**Flowchart af AK47 delen af shoot**



Koden starter ved at spilleren trykker på venstre museknap, derefter tjekkes der om de 4 parametre er sande, hvis de ikke er det stopper scriptet. Hvis de er sande starter den shootcooldown, fjerner et skud fra magasinet og instantiater et skud.

### void Reload



Reload scriptet er delt op i to dele, da det tager længere tid at reloade når våbnet ikke har et skud i sig,

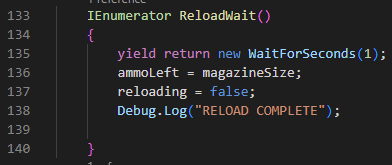
115 Er reload hvor der er skud i magasinet, det tjekkes med ammoLeft > 0. Da man ikke skal kunne reloade når magasinet i fuldt tjekkes det med ammoLeft < magazineSize. For ikke at kunne reloade 2 gange på samme tid tjekkes der også at reloading = false

117 reloading bliver sat til true, da reloading er i gang

118 en coroutine startes, da man ikke kan vente i et void

123 det tjekkes at der er 0 ammunition tilbage i magasinet og at realoding ikke er i gang

### IEnumerator ReloadWait



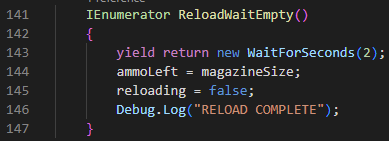
133 Coroutinen kaldes ReloadWait() bliver startet

135 WaitForSeconds bruges til at holde 1 sekunds pause inden den går videre til næste linje

136 magasinet fyldes ved at sætte mængden af amonution til at være størrelsen på magasinet

137 Reloading sættes til at være false da den er færdig med at lade

### IEnumerator ReloadWaitEmpty



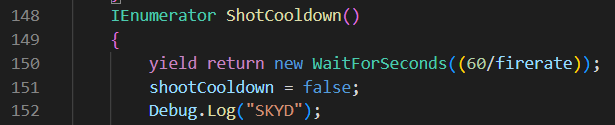
141 Coroutinen kaldes ReloadWaitEmpty() bliver startet

143 WaitForSeconds bruges til at holde 2 sekunds pause inden den går videre til næste linje

144 magasinet fyldes ved at sætte mængden af ammunition til at være størrelsen på magasinet

145 Reloading sættes til at være false da den er færdig med at lade

### IEnumerator ShotCooldown



For at våbnet ikke skyder hvert frame bruges ShotCooldown()

150 Der ventes i 60/firerate for at den skyder med firerate skud i minuttet. Firerate er en public variablen og kan ændres i unity til at være mere passende.

151 Når den har ventet sættes shootCooldown til at være false

## Diskussion

Efter tests af spillet følte vi at det var blevet lidt for nemt. I stedet for bare at sætte sværhedsgraden op, indsatte vi forskellige sværhedsgrader. De forskellige sværhedsgrader gjorde spillet mere interessant, men tilføjede også mulighed for at en mere avanceret spiller også kunne nyde spillet. Sværhedsgraden ændrer bl.a. hvor meget liv spilleren har, og hvor meget liv zombierne har. Ændringerne i sværhedsgraden gør at alle har mulighed for at deltage, derved kan den også vække yderligere interesse for zombiegenren. For yderligere at vække interesse for zombiegenren kunne vi videreudvikle spillet for at gøre det mere interessant, vi har mange ideer til overs fra vores brainstorm, vi startede med at lave. En af vores ideer var en buystation, hvor man for points kunne købe bedre våben eller bedre ammunition. En anden ide var et powerup system, hvor en der var en chance for at en zombie lagde en powerup på jorden når den blev dræbt, powerupsne kunne fx. være uendelige skud i en periode, at man ikke kunne tage skade, eller at man kunne løbe utroligt hurtigt. Andre ideer var stamina og forskellige typer zombier, altså nogle der kunne kravle, nogle der var langsomme med meget HP og lignende. Der kunne også lave forbedringer i UI’et ved at gøre det pænere eller ved at tilføje et highscore system

## Konklusion

VI kan oven på alt dette konkludere at vi har fået lavet et FPS zombie-spil, målrettet til en mindre avanceret målgruppe. I udviklingen har jeg haft fokus på playercontrolleren, UI og våben.

## Grafik credits

Vi har valgt at fokuserer på koden og spillet, derfor har vi fundet graphics’ne på “Unity assets store” da vi ikke mente at det gav mening at lægge vores fokus på at lave objekter fra bunden. Mappet er lavet af “Supercyan” og hedder “Environment Pack: Free Forest Sample”[[1]](#footnote-1). Zombierne er lavet af “Supercyan” og hedder “Character Pack: Zombie Sample”[[2]](#footnote-2). Våbnene er lavet af “Simon Mercuzot” og hedder “Low Poly Weapons VOL.1”[[3]](#footnote-3). Playeren er lavet af “Supercyan” og hedder “Character Pack: Free Sample”[[4]](#footnote-4).

## 

1. <https://assetstore.unity.com/packages/3d/vegetation/environment-pack-free-forest-sample-168396> [↑](#footnote-ref-1)
2. <https://assetstore.unity.com/packages/3d/characters/humanoids/fantasy/character-pack-zombie-sample-131604> [↑](#footnote-ref-2)
3. <https://assetstore.unity.com/packages/3d/props/guns/low-poly-weapons-vol-1-151980> [↑](#footnote-ref-3)
4. <https://assetstore.unity.com/packages/3d/characters/humanoids/character-pack-free-sample-79870> [↑](#footnote-ref-4)