Vincent Creveld 3019866
GDV2

Concept:
Bubble Bobble met weapon select.

Twist:
Weapon/bubble select.
In het originele spel heb je maar 1 soort bubbel waarmee je je enemy af probeert te maken. Deze bubbel schiet maar een kleine afstand. De wapens die ik er zelf in gezet heb is een sniper bubbel, een kleine bubbel met lage fire-rate die meerdere enemies kan raken en een 3-round burst bubble, die 3 bubbels in successie afvuurt.

Mechanics:
De speler zal rondlopen met de W, A, S, D en spatie toetsen. Het wisselen van wapens kan met de 1, 2 en 3 toetsen gedaan worden
De speler schiet door te klikken.

Design patterns:
Observer pattern: enemies stoppen met lopen wanneer de speler dood is.

Event: Enemies die sneller gaan lopen wanneer andere enemies binnen dat level sterven
Wanneer de speler dood gaat gebeurt er van alles binnen de game manager, zoals het afnemen van een leven en het opnieuw opstarten van een level.

Singleton: Level controllers en de game initialiser die sound afspeelt

Scripts (allemaal .cs):
DontDestroy
Enemy
EnemyManager
GameManager
Interfaces
LevelSelect
NextScene
ObjectPool
OneWayPlatform
PlayerController
TeleportBlock
Bubble
ARBubble (child van Bubble)
SniperBubble (child van Bubble)

Unity functionaliteiten:
Rigidbody
UI
Input functies
OnTriggerEnter
OnTriggerStay
OnTriggerExit
OnCollisionEnter
OnCollisionExit
Physics.IgnoreCollision
Default Unity functies (Awake, Start, Update, LateUpdate, FixedUpdate etc…)

Verantwoording keuze van Bubble Bobble:
Voor deze opdracht heb ik voor Bubble Bobble gekozen omdat ik dit spel vroeger erg veel gespeeld heb. Hiernaast dacht ik dat niemand anders voor dit spel zou kiezen. (Pac-man en Snake waren te voor-de-hand liggend).
Ook dacht ik dat Bubble Bobble maken een haalbaar doel was.

Ik heb voor het singleton pattern gekozen om makkelijk vanuit de Player de audiosources in de DontDestroy class te kunnen beinvloeden. Dit had ook met een event gekund, maar dan zou de DontDestroy een referentie moeten hebben naar de speler, en de DontDestroy zou het dan elk nieuw level de speler moeten vinden. Ik heb dus gekozen voor het singleton pattern gekozen om deze interactie efficiënter te maken.

De event/observer patterns zijn gebruikt om 2-richtings references te voorkomen.

Veranderingen tegenover het originele ontwerp:
Ik heb de beslissing genomen om de enemies uiteindelijk niet in een object pool te gooien omdat de levels premade zijn. Er is dus geen reden om de enemies herhaaldelijk in te moeten spawnen. Als vervanging om toch te kunnen detecteren wanneer er maar een enemy over is of wanneer het level cleared is heeft een parent empty object een scriptje gekregen waar simpelweg de children geteld worden. De parent is dan ook gesubscribed op de enemyDeath events binnen de children. De functie om de children te bekijken wordt dan ook alleen maar aangeroepen wanneer een Enemy gedood wordt door de speler om niet elke frame te hoeven checken.

Ik heb besloten om uiteindelijk geen score bij te houden, aangezien het spel gebaseerd is op levels clearen in plaats van oneindige enemies in te blijven spawnen. Het bijhouden van score zou dus altijd op dezelfde waardes uitkomen. Dus onafhankelijk van hoe de speler het level uitspeelt, zal de score er hetzelfde uit zijn gekomen.

Verbeteringen aanpak volgende keer/spel in het algemeen:
In het vervolg zou ik de enemies en speler class laten inheriten van dezelfde base class. Ik zou dit willen doen omdat beide classes veel overeenkomende functies hebben. Ik wil nog meer variaties op enemies en wapens willen hebben in een vervolg.

Ik ben dus niet aan het maken van meer wapens en enemies toegekomen.

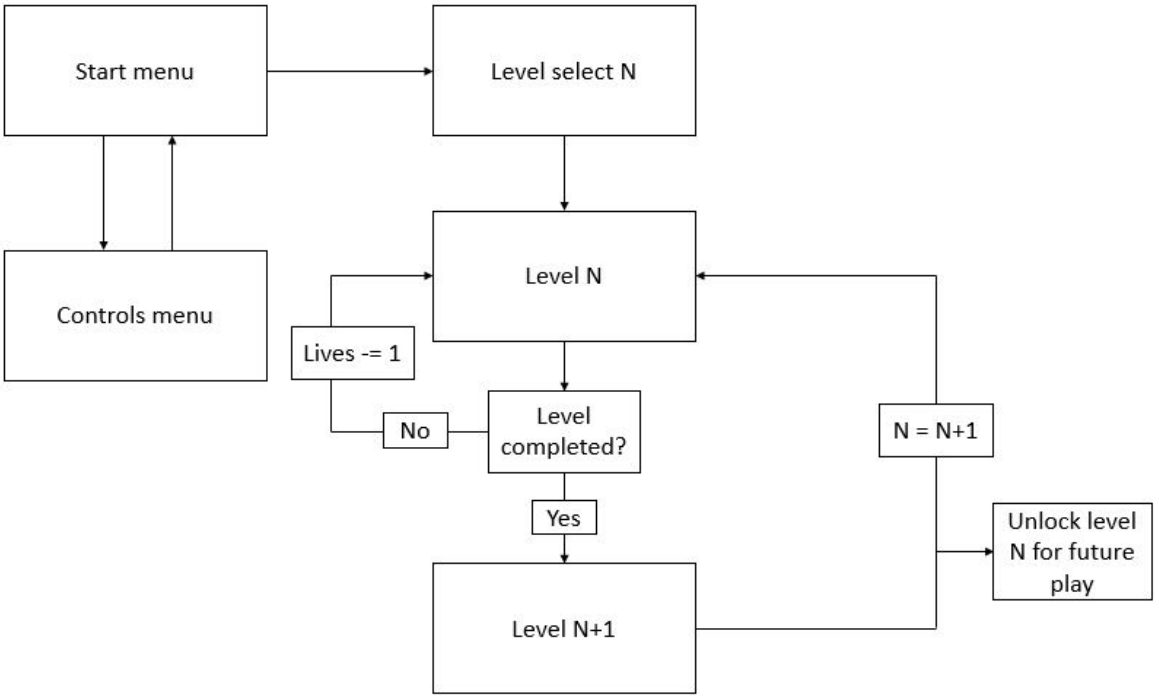
In het vervolg ga ik ook de originele sprite sheet gebruiken voor het spel. Ik kwam deze keer te laat er achter dat die spritesheets online beschikbaar zijn.

Toelichting UML:
Zoals te zien is, is bijna elke class derived van MonoBehaviour.

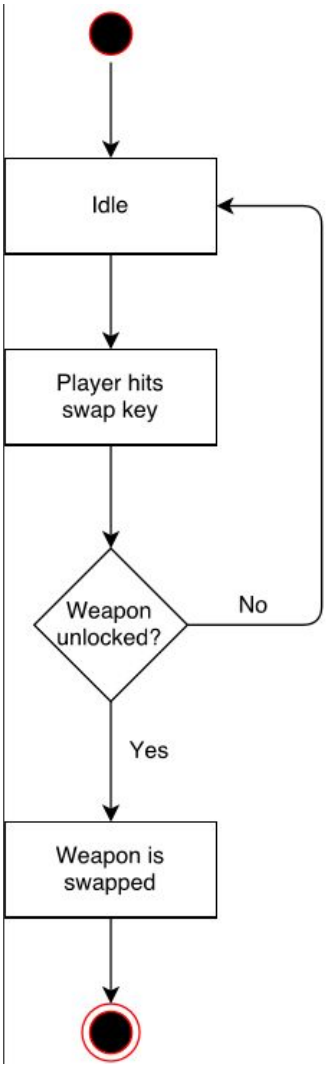
De PlayerController en Enemy classes implementeren beide de ICanPass interface. Deze interface is leeg en wordt puur gebruikt als een secundaire tag waarmee de OneWayPlatform class kan inzien wie het door kan laten en wat allemaal niet.

De GameManager heeft een referentie naar de PlayerController in de class om makkelijk bij de events binnen de PlayerController te komen. Dit is om te luisteren naar de playerDeath.

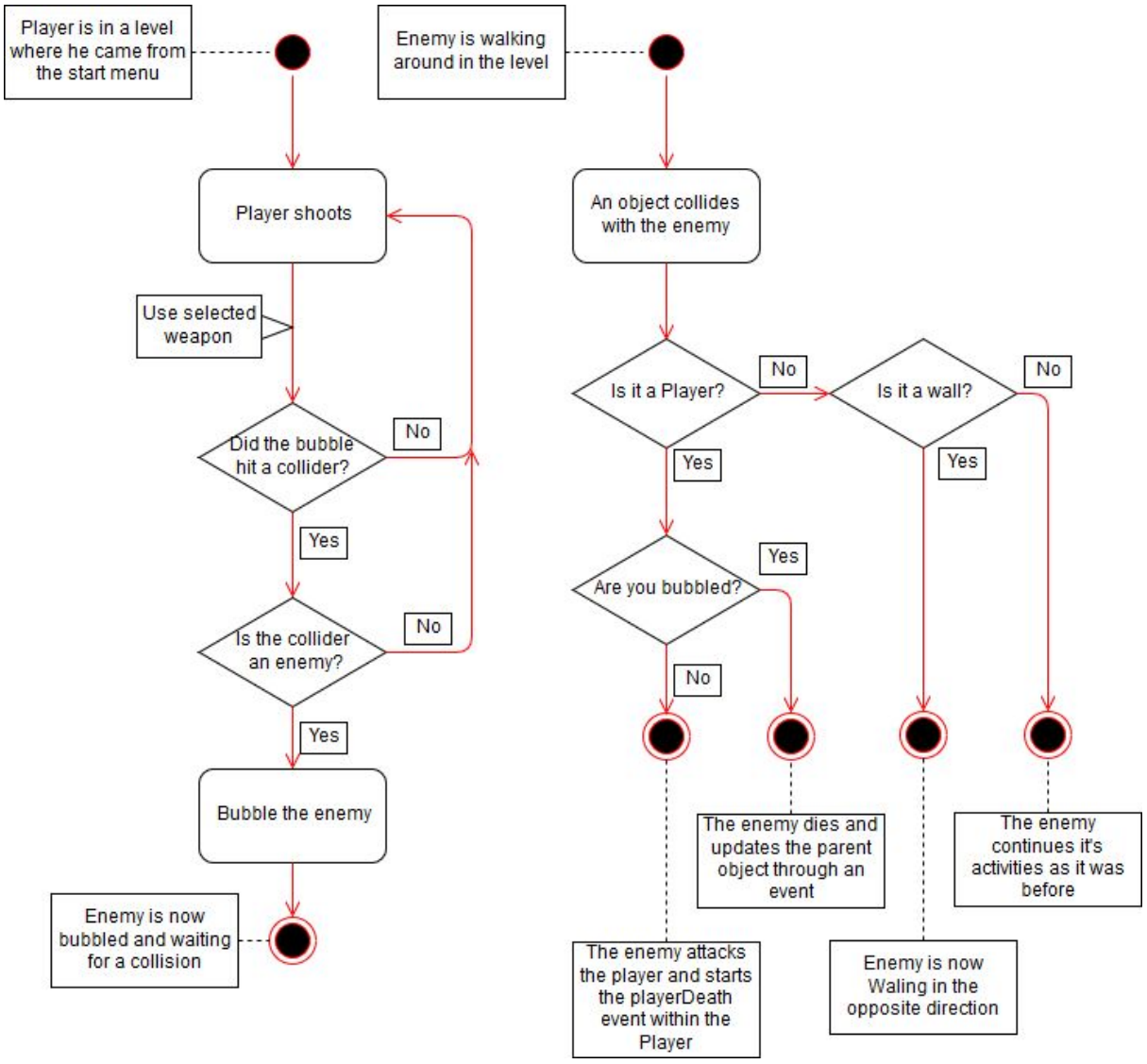
Sceneflow:


Start menu
Level select N
Controls menu
Level N
Lives -= 1
No
Level completed?
Yes
N = N+1
Unlock level N for future play
Level N+1

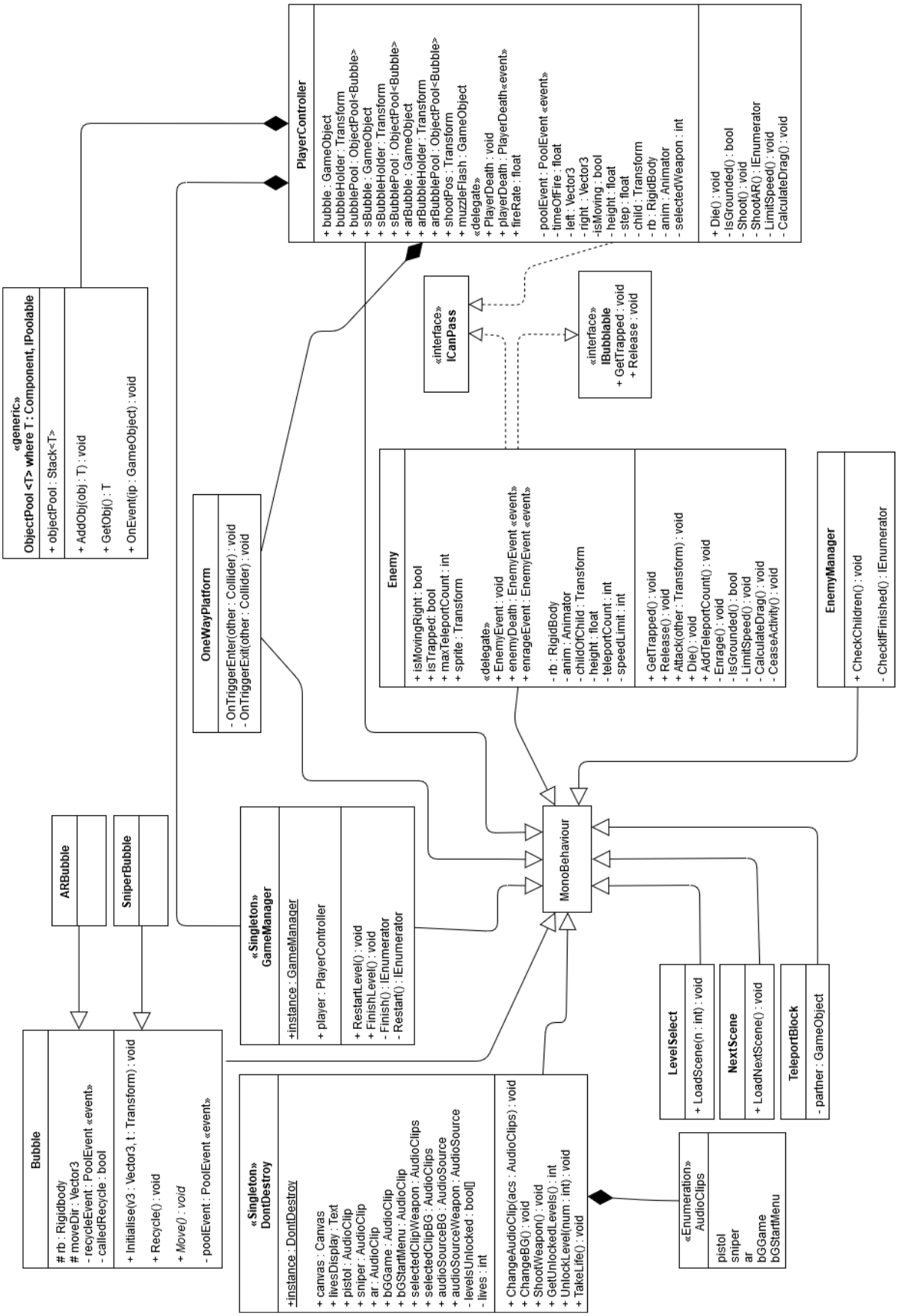
Activity Diagram weapon switch:



Activity Diagram spel:



UML spel:


«generic»
ObjectPool <T> where T : Component, IPoolable
+ objectPool : Stack<T>
+ AddObj(obj : T) : void
+ GetObj() : T
+ OnEvent(ip : GameObject) : void
PlayerController
+ bubble : GameObject
+ bubbleHolder : Transform
+ bubblePool : ObjectPool<Bubble>
+ sBubble : GameObject
+ sBubbleHolder : Transform
+ sBubblePool : ObjectPool<Bubble>
+ arBubble : GameObject
+ arBubbleHolder : Transform
+ arBubblePool : ObjectPool<Bubble>
+ shootPos : Transform
+ muzzleFlash : GameObject
«delegate»
+ PlayerDeath : void
+ playerDeath : PlayerDeath«event»
+ fireRate : float
- poolEvent : PoolEvent «event»
- timeOfFire : float
- left : Vector3
- right : Vector3
- isMoving : bool
- height : float
- step : float
- child : Transform
- rb : RigidBody
- anim : Animator
- selectedWeapon : int
+ Die() : void
- IsGrounded() : bool
- Shoot() : void
- ShootAR() : IEnumerator
- LimitSpeed() : void
- CalculateDrag() : void
«interface»
ICanPass
«interface»
IBubblable
+ GetTrapped : void
+ Release : void
OneWayPlatform
- OnTriggerEnter(other : Collider) : void
- OnTriggerExit(other : Collider) : void
Enemy
+ isMovingRight : bool
+ isTrapped: bool
+ maxTeleportCount : int
+ sprite : Transform
«delegate»
+ EnemyEvent : void
+ enemyDeath : EnemyEvent «event»
+ enrageEvent : EnemyEvent «event»
- rb : RigidBody
- anim : Animator
- childOfChild : Transform
- height : float
- teleportCount : int
- speedLimit : int
+ GetTrapped() : void
+ Release() : void
+ Attack(other : Transform) : void
+ Die() : void
+ AddTeleportCount() : void
- Enrage() : void
- IsGrounded() : bool
- LimitSpeed() : void
- CalculateDrag() : void
- CeaseActivity() : void
EnemyManager
+ CheckChildren() : void
- CheckIfFinished() : IEnumerator
ARBubble
SniperBubble
«Singleton»
GameManager
+instance : GameManager
+ player : PlayerController
+ RestartLevel() : void
+ FinishLevel() : void
- Finish() : IEnumerator
- Restart() : IEnumerator
MonoBehaviour
Bubble
rb : Rigidbody
moveDir : Vector3
- recycleEvent : PoolEvent «event»
- calledRecycle : bool
+ Initialise(v3 : Vector3, t : Transform) : void
+ Recycle() : void
+ Move() : void
- poolEvent : PoolEvent «event»
«Singleton»
DontDestroy
+instance : DontDestroy
+ canvas : Canvas
+ livesDisplay : Text
+ pistol : AudioClip
+ sniper : AudioClip
+ ar : AudioClip
+ bGGame : AudioClip
+ bGStartMenu : AudioClip
+ selectedClipWeapon : AudioClips
+ selectedClipBG : AudioClips
+ audioSourceBG : AudioSource
+ audioSourceWeapon : AudioSource
- levelsUnlocked : bool[]
- lives : int
+ ChangeAudioClip(acs : AudioClips) : void
+ ChangeBG() : void
+ ShootWeapon() : void
+ GetUnlockedLevels() : int
+ UnlockLevel(num : int) : void
+ TakeLife() : void
«Enumeration»
AudioClips
pistol
sniper
ar
bGGame
bGStartMenu
LevelSelect
+ LoadScene(n : int) : void
NextScene
+ LoadNextScene() : void
TeleportBlock
- partner : GameObject