Technische Documentatie Level Editor Tool
Kernmodule Game Development Blok 2

Jelle Dekkers GDV2

CONTENTS

INLEIDING

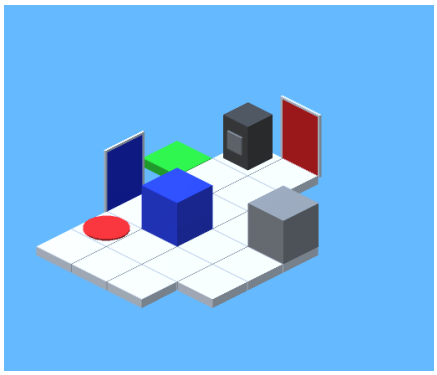
Dit is het technische document voor de Level Editor Tool van Jelle Dekkers gemaakt voor de KernModule Game Development in Blok 2 schooljaar 17/18. In dit document wordt onder andere uitgelegd wat de tool doet, hoe het gebruikt wordt en hoe het werkt.

WAT IS HET?

De tool is een Level Editor waarbij gemakkelijk tegels, blokjes en muren neergezet kunnen worden om zo een level te maken.

Gemaakte levels worden omgezet naar XML files die makkelijk opgeslagen en uitgelezen kunnen worden.

WAAROM



Ik ben al een tijdje bezig met het maken van een spel. In dit spel bestuurt de speler een blokje die over tegels kan lopen met als doel om de groene tegel te bereiken. Gaandewegs moet de speler puzzles oplossen en obstakels ontwijken in de vorm van andere blokken en speciale tegels. Ook zijn er portalen die de speler kan gebruiken om zijn doel te bereiken.

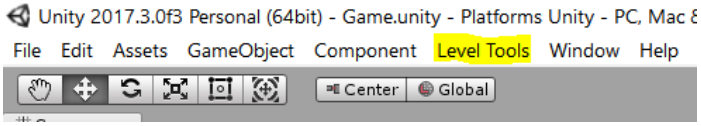
Ik ben van plan om in het uiteindelijke spel veel levels te hebben(denk aan spellen zoals Angry Birds die ook veel levels bevatten). Om al deze levels als Unity Scenes te builden zal veel onnodige ruimte in beslag genomen worden. Door levels te serialiseren als XML bespaar ik veel ruimte.

Ik heb deze tool gemaakt om 3 redenen:

- Om het maken van levels sneller te maken
- Om het maken van levels is makkelijker te maken
- Om ruimte te besparen door XML files te gebruiken i.p.v. Unity scenes

HOE GEBRUIK JE HET?

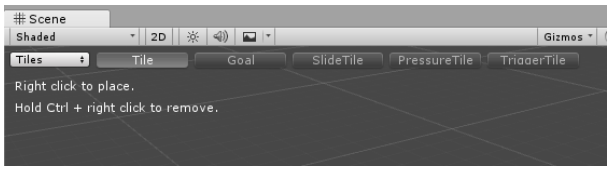
1. Druk op Level Tools in de menubalk in Unity en klik op Level Builder om de Tool aan of uit te zetten



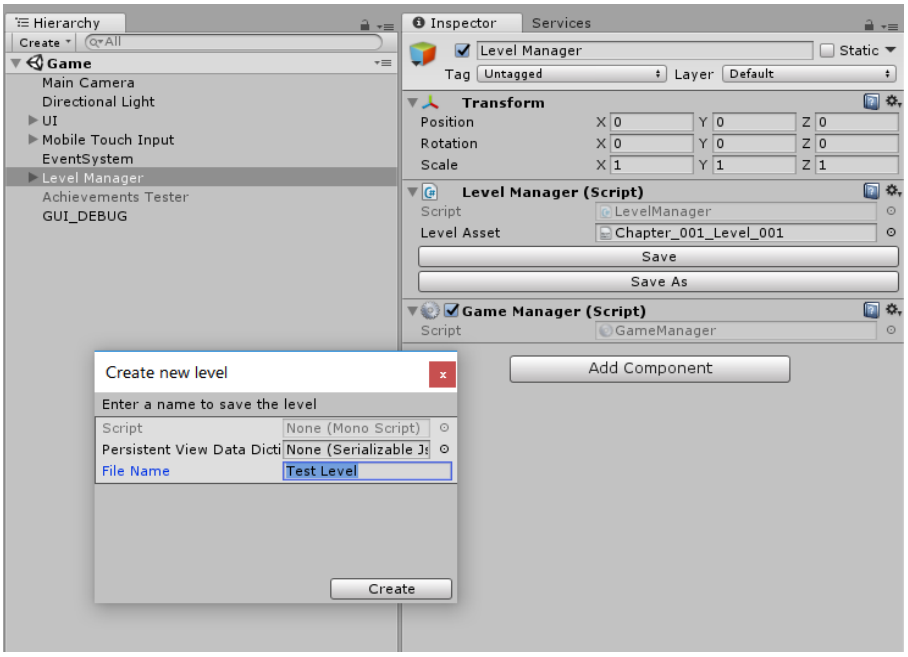
2. Kies een bouw modus(Tiles, Blocks of Walls)

3. Selecteer de prefab die je wilt neerzetten

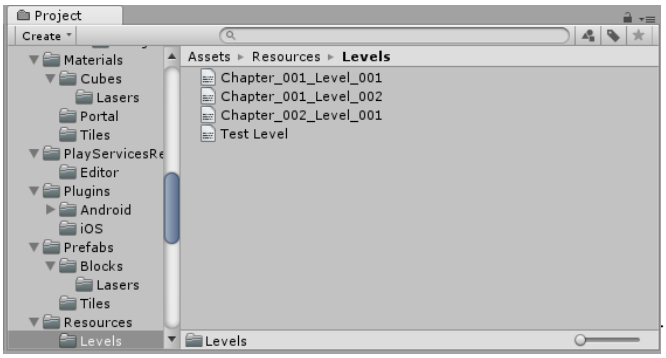
4. Klik op de plek in de scene waar je de gekozen prefab wilt plaatsen(als het plaatsbaar is dan is de wireframe groen, anders rood). Door 'ctrl' ingedrukt te houden kan worden gepaint.



5. Op de Level manager zit een Level Asset variabele. Als de ze null is dan zal alleen de optie 'Save As' beschikbaar zijn. Hiermee opent een klein schermpje waar de naam van de nieuwe file kan worden ingevuld. Hierna wordt een nieuwe file aangemaakt en in de Resources/Levels folder gezet. Als er wel een level Asset variabele aanwezig is dan kan deze worden opgeslagen door op 'Save' te drukken. Het opslaan van levels is alleen mogelijk als er tenminste 1 player block en 1 goal tile aanwezig zijn. Dit is zo gemaakt om er voor te zorgen dat alle levels speelbaar zijn. Is dit niet het geval dan zijn de save knoppen doorzichtig en verschijnt er een tooltip als de muis op 1 van de knoppen staat.

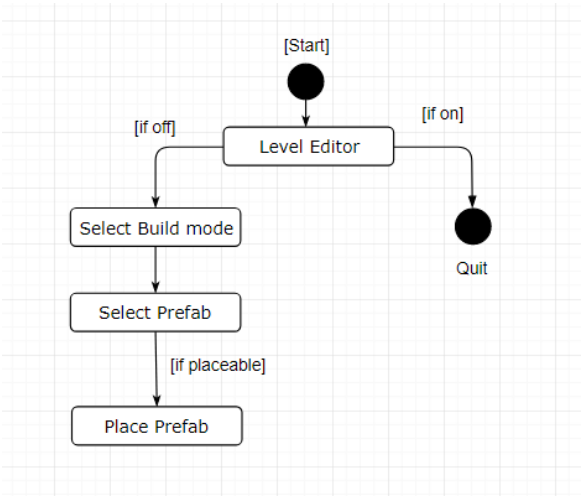


6. Sleep een opgeslagen XML file in de Level Asset variabele en deze wordt automatisch in de scene gebouwd en is direct aan te passen.

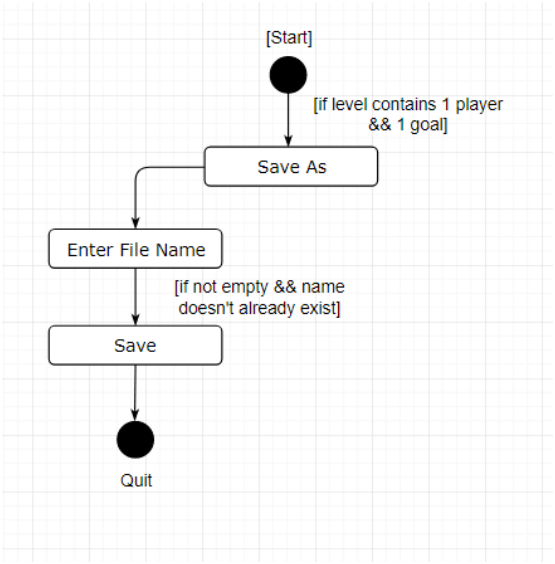


ACTIVITY DIAGRAM

Hieronder staat de Activity Diagram voor het creëeren van een level.



Hieronder staat de Activity Diagram voor het opslaan van een level.



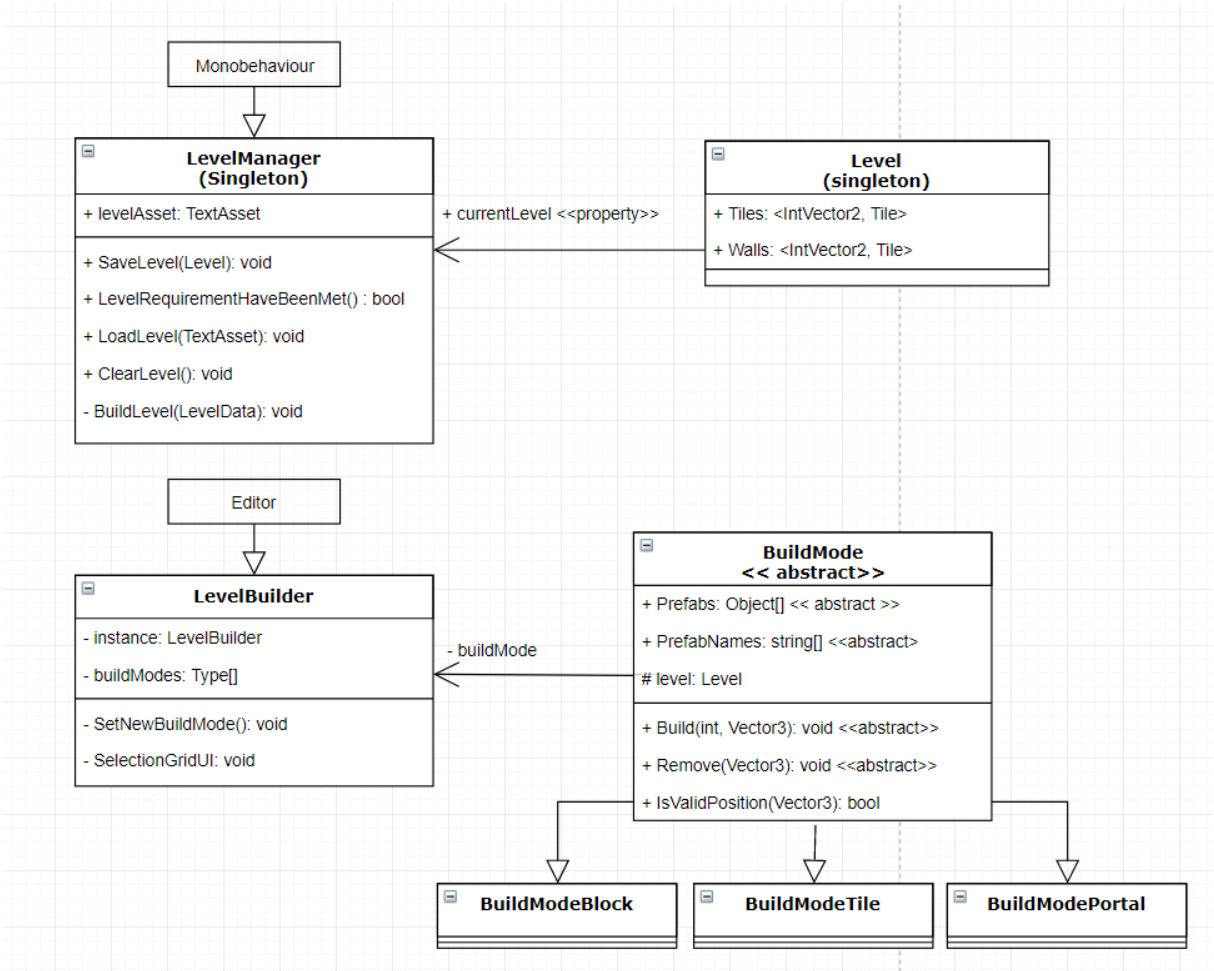
CODE STRUCTUUR

Een level class heeft een Dictionary voor tiles. Bij het plaatsen van een tile wordt er een entry toegevoegd bij de tile dictionary. Hetzelfde gebeurd voor walls(portals). Blocks worden opgeslagen per tile en kunnen in de scene dan ook alleen geplaatst worden op tiles.

LevelManager.cs heeft een currentLevel variabele van het type Level en heeft functies voor het Saven, Loaden en het opbouwen van levels.

Bij het opslaan wordt de currentLevel omgezet naar levelData.cs. LevelData bevat de benodigde logica om alles Level om te zetten naar serialiseerbare data en bevat variabelen voor TileData, BlockData en PortalData. Deze LevelData wordt vervolgens geserialiseerd naar XML en de file wordt in de Resources/Levels folder gezet.

Bij het deserialiseren wordt er gekeken naar de ScriptableObject genaamd PrefabManager. Deze staat in Resources en heeft een dictionary van key Type en Value GameObject voor Tiles, Blocks en Walls. Hiermee kan makkelijk bij het deserialiseren de correcte prefab worden gepakt die past bij de gedeserialiseerde Type.



CONCLUSIE

Het maken van een Level Editor was voor mijn project extreem handig. Aangezien tiles worden opgeslagen in een dictionary zou het onbegonnen werk zijn om dit handmatig, via de inspector, te moeten doen.

Het implementeren van object painting bespaart mij ook immens veel tijd omdat ik dus niet steeds geplaatste prefabs steeds moet kopieren en plaatsen(en de correcte variabele te vullen).

Ook bespaart het enorm veel ruimte in het geheugen. Mijn Unity scene met level objecten is rond de 92kb(in de editor en dus niet in een build), een XML level is rond de 3kb. Dit bespaart dus heel veel ruimte, vooral als er veel levels zijn.

De tijd die ik heb gestopt in het maken van deze tool is eigenlijk niets vergeleken met de tijd die ik uiteindelijk zal besparen met het maken van levels.

VERBETERINGEN

Eventuele verbeteringen die ik nog graag had willen toepassen was een mooier uitziende UI in de sceneview voor het selecteren van prefabs, helaas had ik hier geen tijd meer voor. Ook was een soort Asset preview leuk geweest als je bijvoorbeeld met de muis boven de knop hangt.