# Module User Experience | Gaming

## Onderdeel Unity Practicum 4: Meer levels

## Auteurs

Paul Bergervoet (Gameplay en teamleiding)

Monique Dewanchand (C#)

Lars Tijsma (Unity)

Anton Visser (Gameplay en testen)



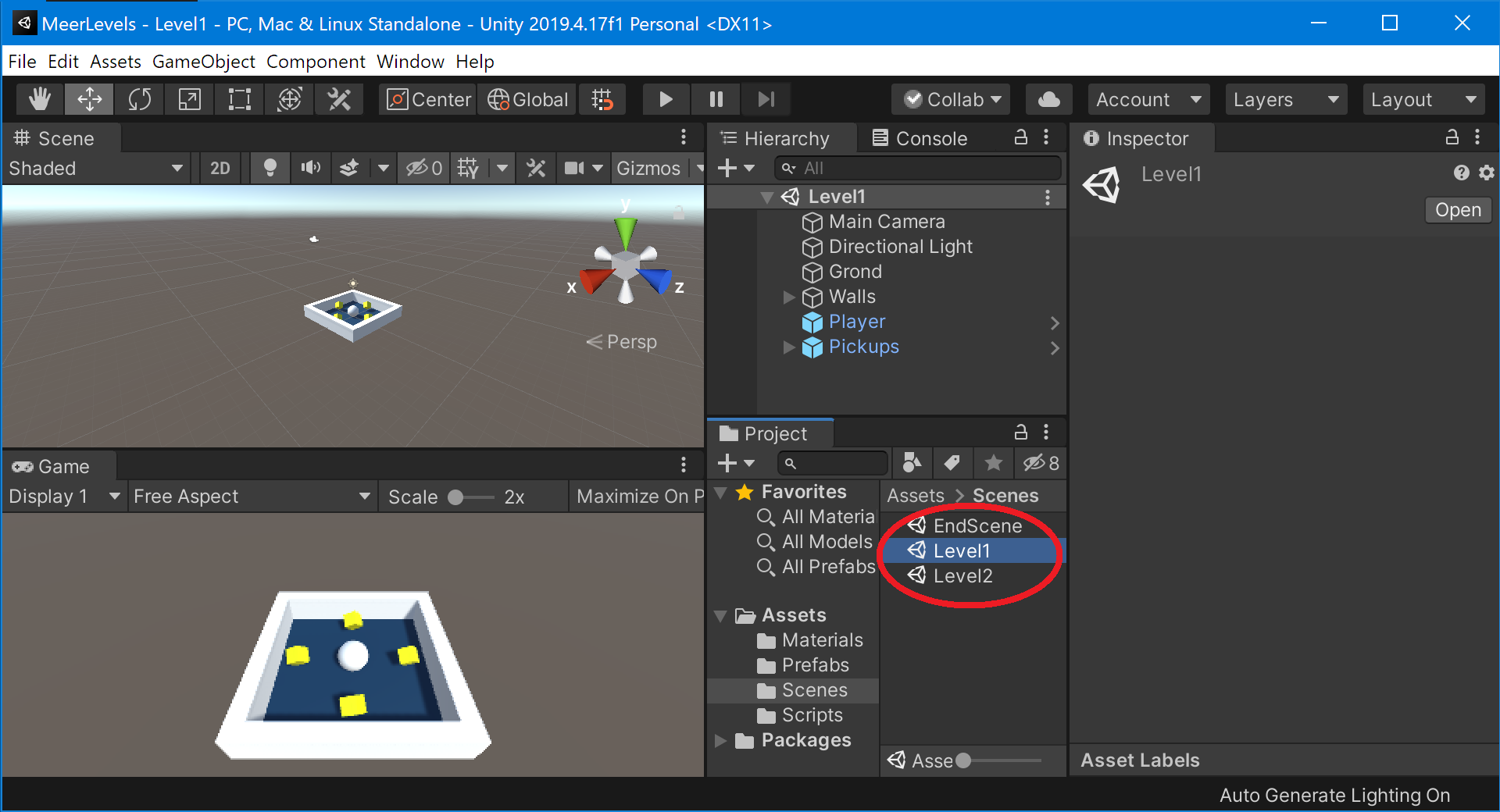
Unity-opdracht bij GD4: Meer Levels

In deze tutorial ga je meerdere levels aan het spel toevoegen en ga je ervoor zorgen dat er tussen de levels gewisseld kan worden.

Bij dit practicum is het handig om het project te gebruiken dat speciaal voor dit practicum gemaakt is. Dit heeft al enkele levels, die alleen nog maar aan elkaar geknoopt hoeven te worden. Je kunt dit project ook gebruiken om te af te kijken hoe je in je eigen game levels kunt maken.

# Setup

* Open het project *MeerLevels* en zorg dat Level1 geopend is in de Unity Editor.
* Bekijk de map *Scenes*. Als het goed is heb je nu drie scenes in de *Scenes* folder van het project.



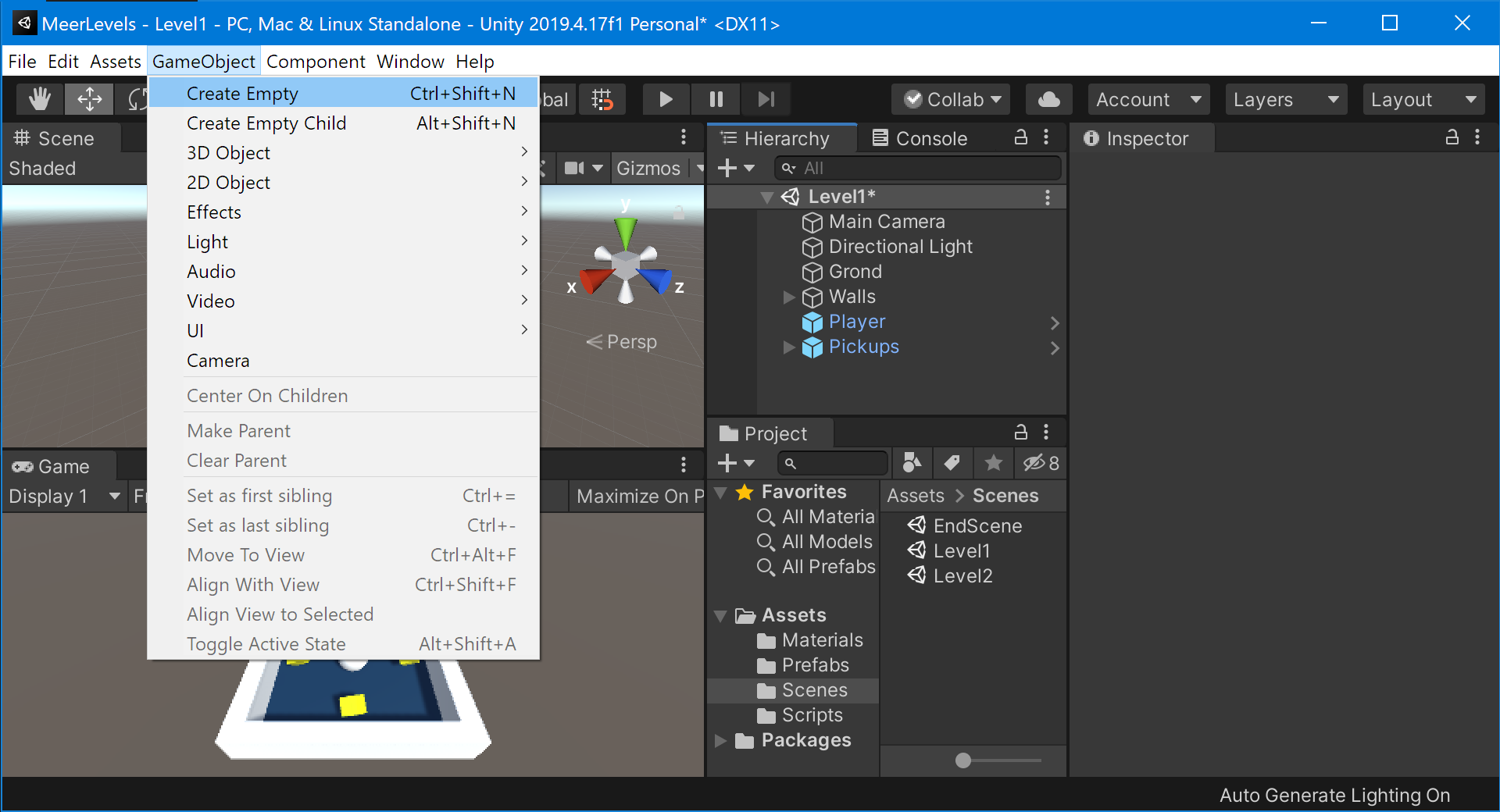
Als je in de map Prefabs kijkt, dan zie je dat we in dit project meer prefabs hebben gemaakt, dan in eerdere projecten. Dit komt doordat we in meerdere scenes dezelfde gameobjecten willen gebruiken en dan is het erg handig om prefabs van deze gameobjecten te hebben.

# Game Manager maken

De code waarmee er tussen scenes gewisseld gaat worden past niet zo goed in de game objecten die er nu al zijn.

Daarom maken we voor dit soort functionaliteit meestal een nieuw onzichtbaar game object die we GameManager noemen.

* Maak een nieuw *empty game object* via het menu *GameObject > Create Empty.*



* Verander de naam van dit nieuwe gameobject in *GameManager*.
* Maak in GameManager een nieuw script met de naam *LevelSwichter.*
* Plaats onderstaande code in het script:

| using System.Collections; using System.Collections.Generic; using UnityEngine; using UnityEngine.SceneManagement;  public class LevelSwitcher : MonoBehaviour {  void Update()  {  if (Input.GetKeyDown(KeyCode.PageUp)) {  SceneManager.LoadScene("Level2");  }   } } |
| --- |

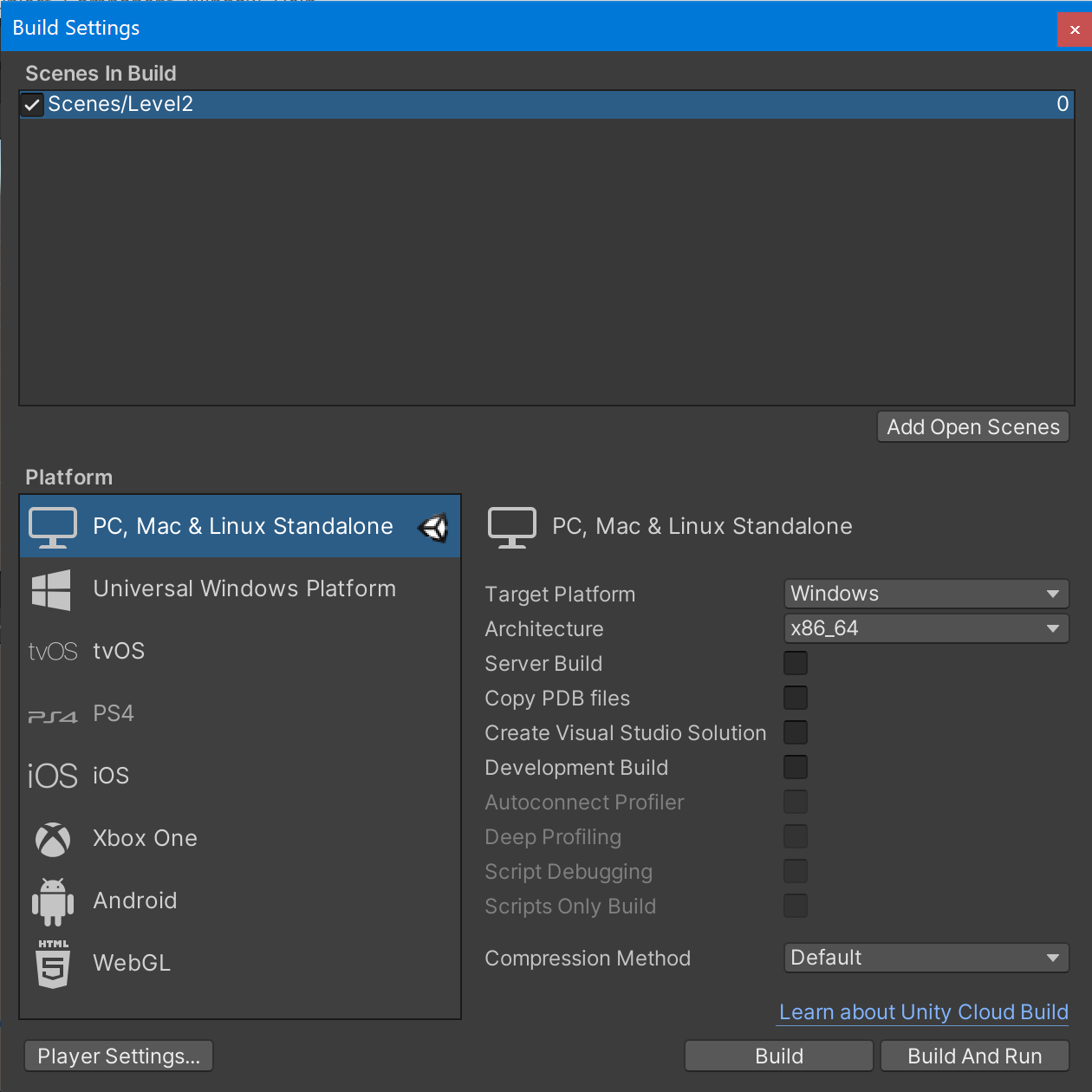
Dit script is niet voor een speler bedoeld, maar voor jou als programmeur, zodat je het spel sneller kunt testen. We kiezen daarom een toets die waarschijnlijk niet in het spel gebruikt gaat worden. Voordat je het spel publiceert is het handig om dit script te deactiveren (of achter een geheime cheat code te verbergen). Hier gaan we verder niet op in.

Test het spel en je ziet onderstaande error

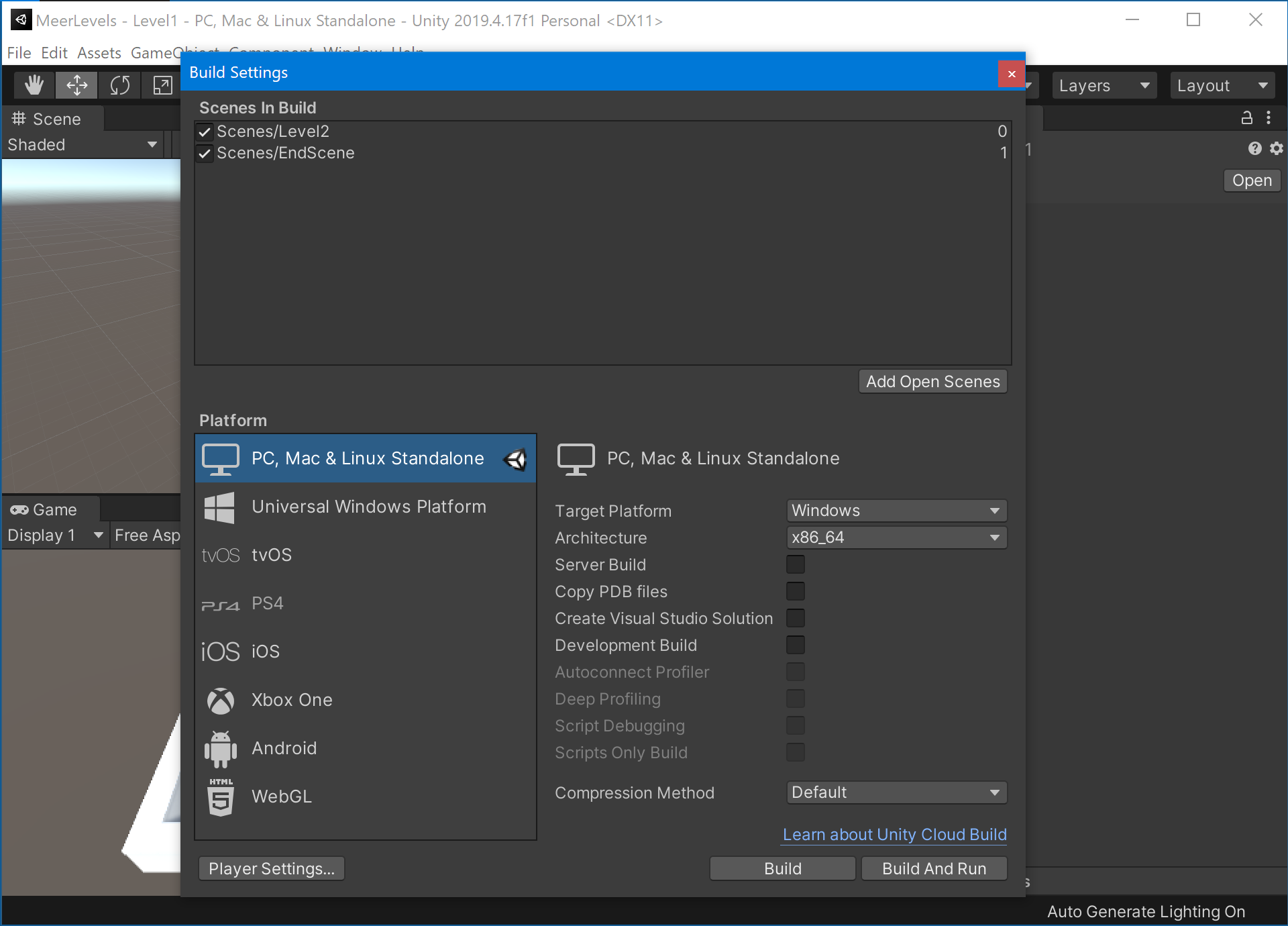
Scene 'Level2' couldn't be loaded because it has not been added to the build settings or the AssetBundle has not been loaded.

Om deze fout op te lossen moet je alle levels die je tijdens het runnen van de game wilt laden toevoegen aan de assetbundle:

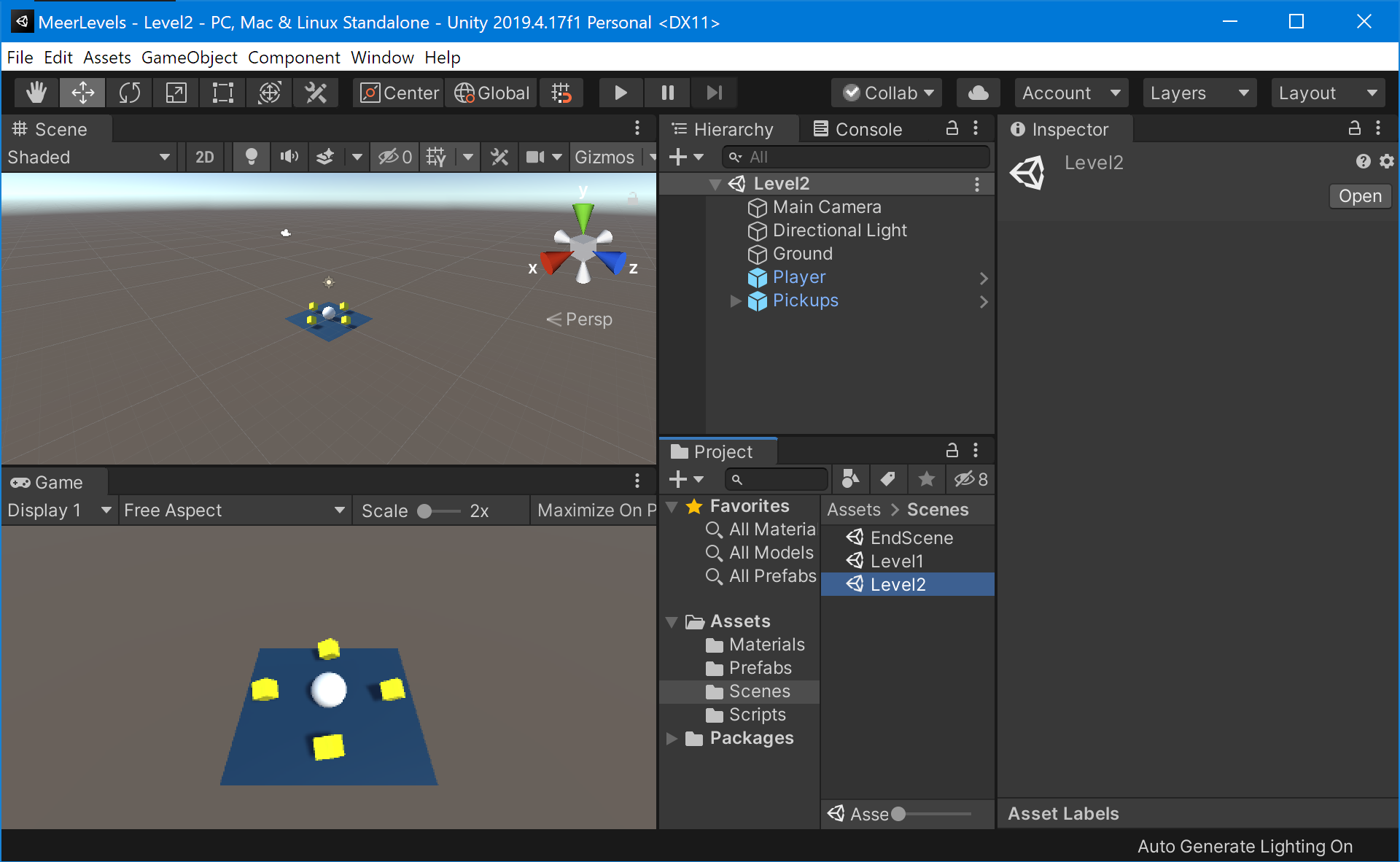
* Zorg dat alle wijzigingen in Level1 opgeslagen zijn en open *Level2* door erop te dubbelklikken.
* Ga naar File > Build Settings en druk op de knop “Add Open Scenes”.
* Klik daarna op het kruisje van het Build Settings venster (dus je hoeft niet op de knop Build of Build And Run te drukken).



* Open nu weer *Level1* en probeer het spel. Als het goed is kun je nu naar *Level2* als je op PageUp drukt.
* Herhaal nu bovenstaande stappen voor de *EndScene* en zorg ervoor dat de build settings er als volgt uitziet:



* Probeer het spel. Als het goed is kun je wel naar *Level2*, maar niet naar *EndScene*.
* Open *Level2* weer in de editor en inspecteer de *Hierarchy*. Als het goed is, zie je de GameManager daar niet tussenstaan.



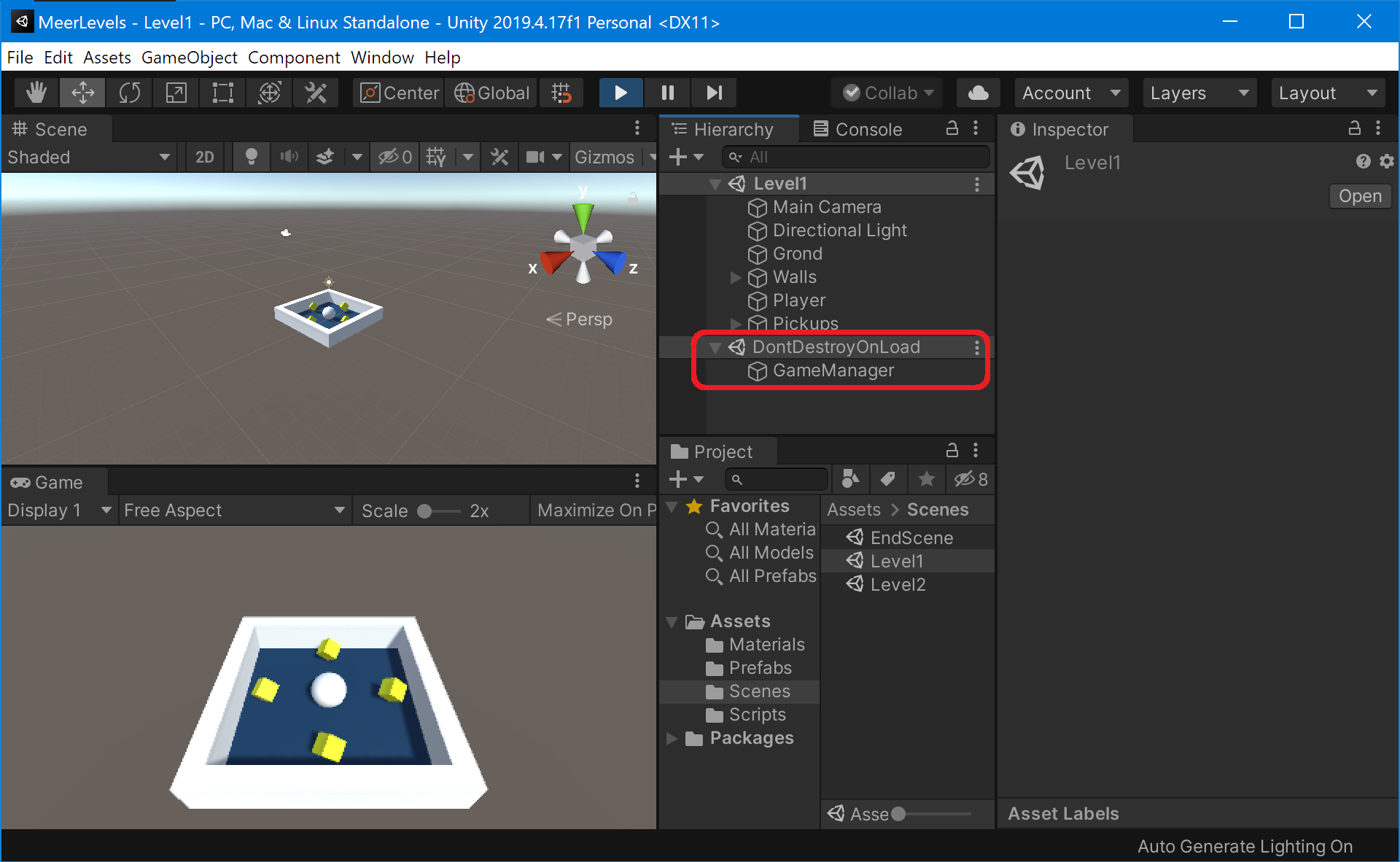
[Afbeelding](05\_geen\_game\_manager\_in\_level2)

Je kunt de GameManager handmatig aan elke scene toevoegen, maar het is makkelijker om de functie DontDestroyOnLoad te gebruiken.

* Voeg onderstaande code toe aan het LevelSwitcher-script.

| void Awake ()  {  DontDestroyOnLoad(this);  } |
| --- |

* Open Level1 weer en test het spel. Als het goed is, staat in de Hierarchy de GameManager nu onder DontDestroyOnLoad zoals hieronder te zien is



* Ga ook naar Level2 en controleer dat de GameManager nog steeds in de Hierarchy staat.
* Probeer ook naar *EndScene* te gaan en als het goed is lukt dit niet.

De code van GameManager kan nu alleen naar Level2. Je kunt aan deze code ook de EndScene toevoegen, maar dan zouden we de code altijd moeten veranderen als er nieuwe levels bijkomen.

Unity houdt zelf een genummerd lijstje bij van alle scenes die in de build settings zijn. Het enige dat het script hoeft te doen is het huidige nummer opvragen en deze met één verhogen.

* Verander de code om level te wisselen als volgt

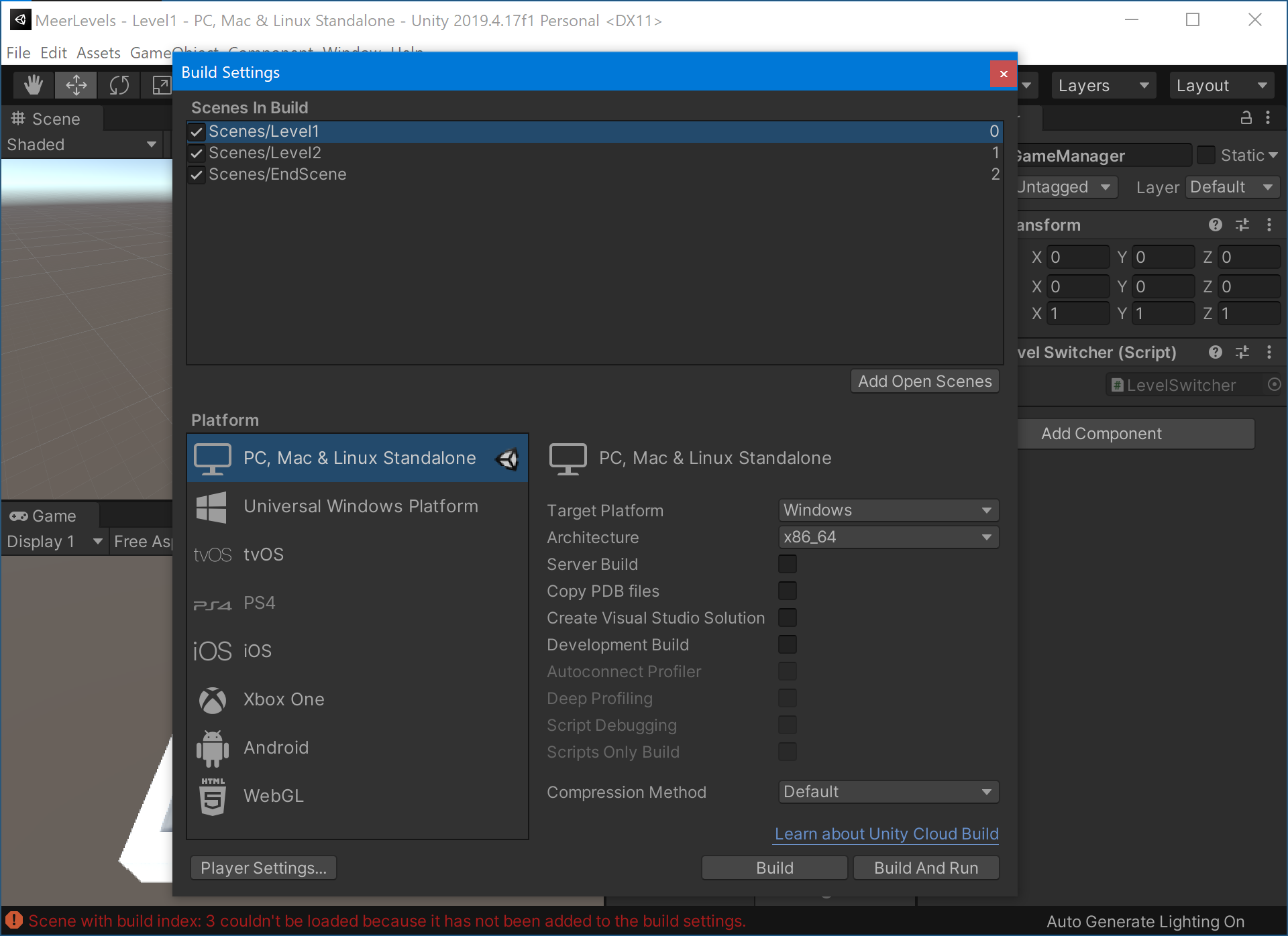
| if (Input.GetKeyDown(KeyCode.PageUp)) {  int huidigeScene = SceneManager.GetActiveScene().buildIndex;  SceneManager.LoadScene(huidigeScene + 1);  } |
| --- |

* Zorg dat Level1 open staat en test het spel. Als het goed is, zie je de volgende foutmelding:

Scene with build index: 3 couldn't be loaded because it has not been added to the build settings.

Het probleem is, dat we Level1 zelf niet aan de build settings hebben toegevoegd waardoor deze door Unity als laatste in de lijst wordt gezet.

* Open de build settings en voeg Level1 toe aan de Scenes In Buid en sleep Level1 helemaal naar boven.



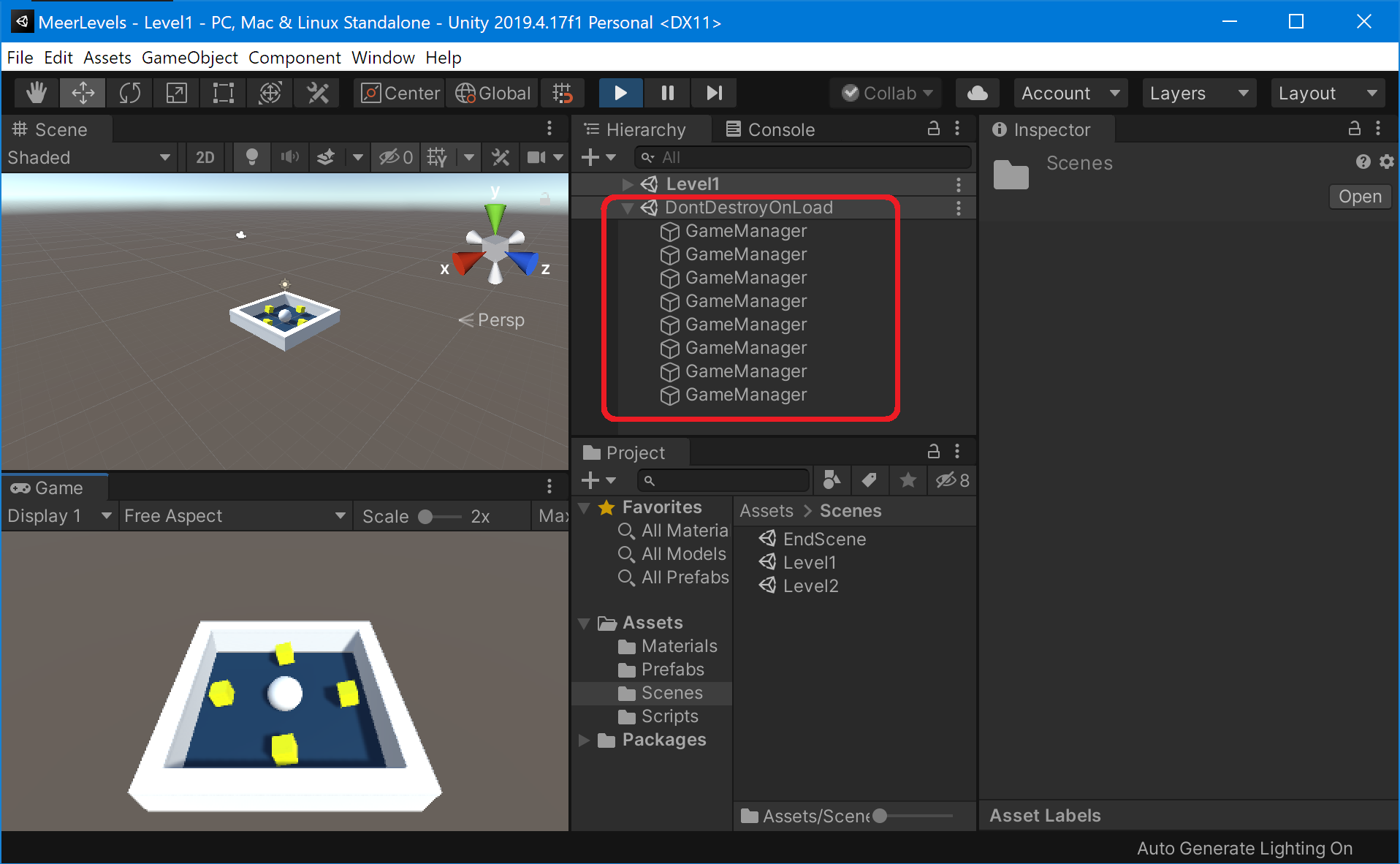
Je kunt overigens het volgnummer dat Unity gebruikt helemaal aan de rechterkant in de lijst zien.

* Sluit de Build Settings en test het spel. Als het goed is kun je nu van *Level1* naar *EndScene* komen.

### Terug naar Level1

Als je in EndScene probeert door te gaan naar het volgende lever, dan krijg je een foutmelding

* Los deze foutmelding op door weer terug naar Level1 te gaan, als je in de EndScene bent.
* Draai het spel en ga een aantal keer door alle scenes heen. Kijk terwijl draait in de Hierarchy en je ziet dat je nu meerdere GameManagers hebt.



#### Vraag

Probeer uit te leggen waardoor het aantal GameControllers verdubbelt elke keer dat je de Level1 laadt?

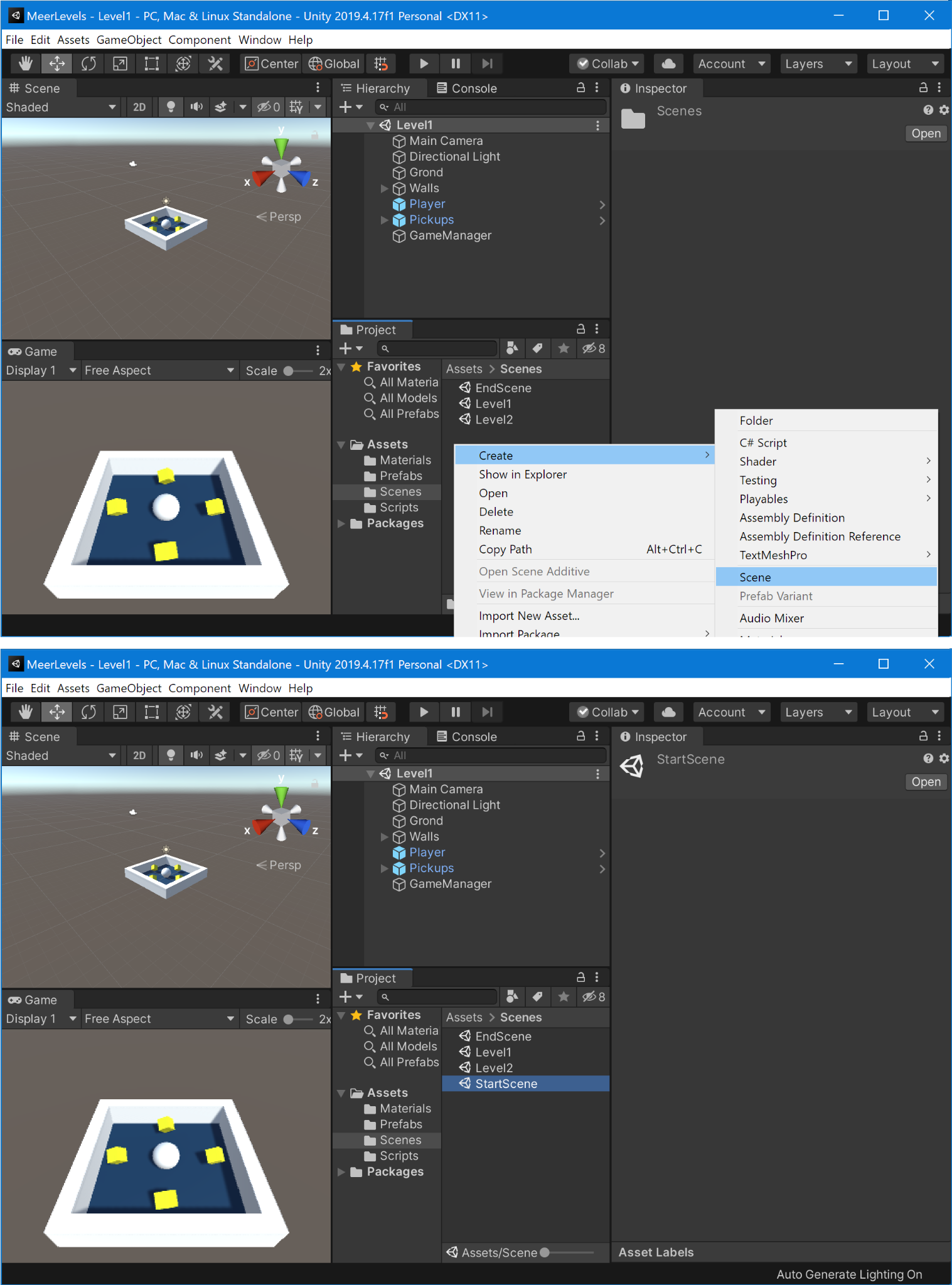
#### Antwoord

Er gebeuren twee dingen. Het eerste effect is dat de eerste scene geladen wordt, wordt er een nieuwe GameController gemaakt omdat deze handmatig is toegevoegd aan deze scene. Het tweede effect is dat elke GameManager die eerder is gemaakt de nieuwe scene opnieuw inlaadt. Dus als er bijvoorbeeld al 2 GameManagers zijn, dan wordt elke scene 2 keer geladen. Elke keer dat Level1 geledan wordt, wordt er een nieuwe GameManager toegevoegd aan DontDestroyOnLoad, dus als we Level1 twee keer laden, eindigen we met 4 GameManagers.

### Startscene maken

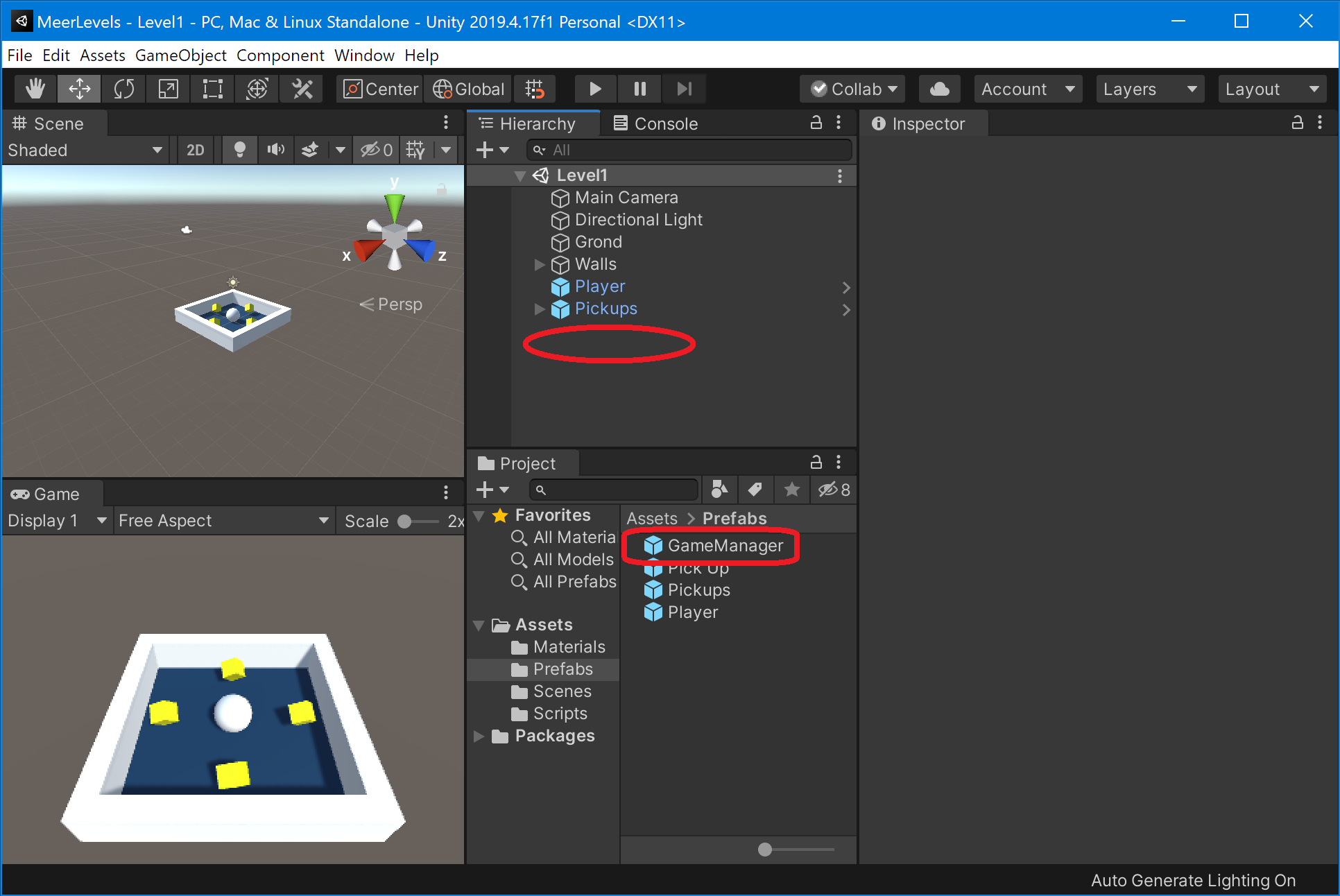
De makkelijkste manier om dit op te lossen, is het maken van een startscene waar je alleen terecht kunt komen als het spel gestart wordt.

* Maak een nieuwe scene en noem deze *Startscene*.

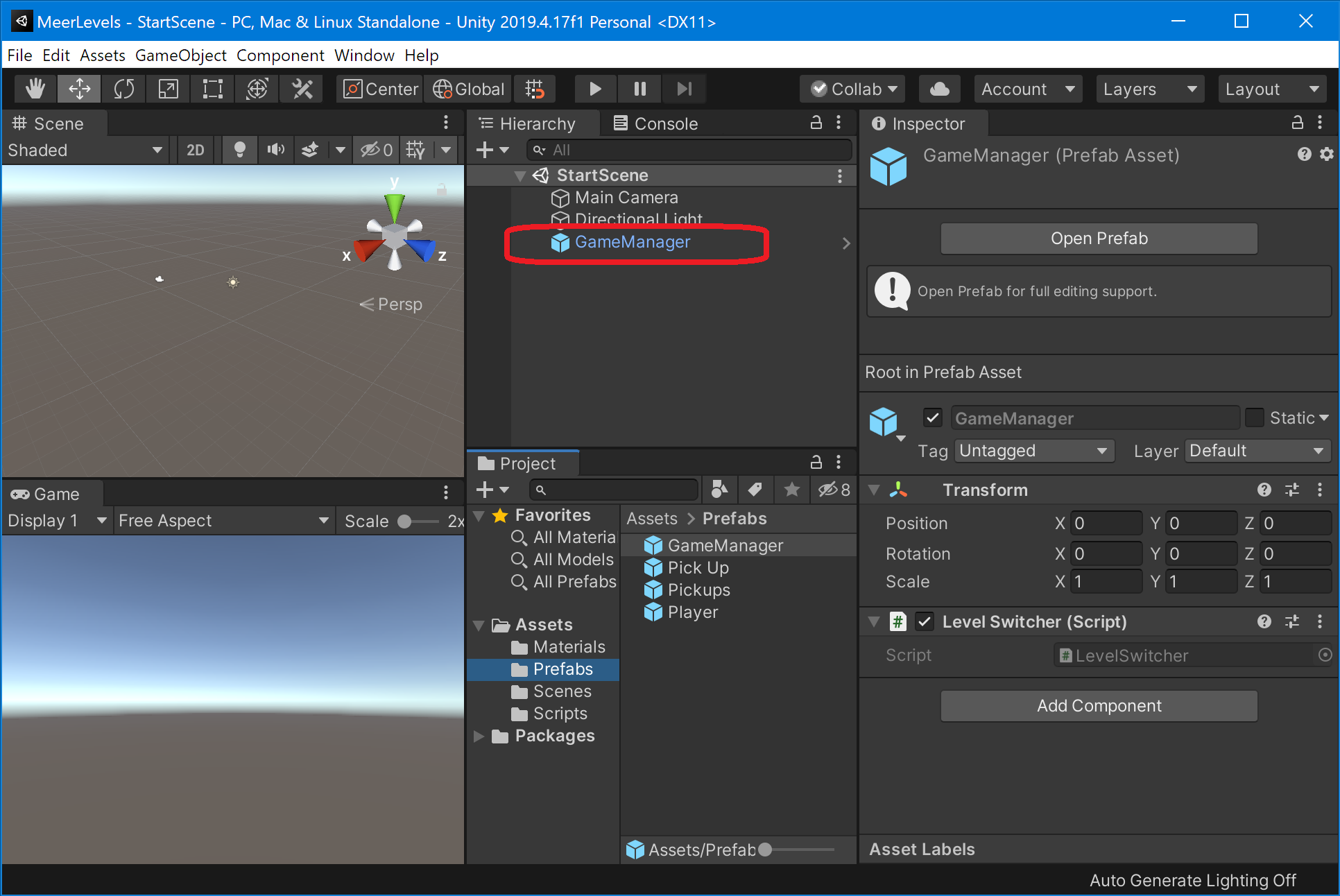


De GameManager moet nu verplaatst worden naar de nieuw gemaakte *StartScene*. Dat doen we door er een prefab van te maken. Hierdoor kunnen we ook de GameManager later aanpassen zonder deze scene open te hebben.

* Sleep de GameManager naar de prefabs-map in het Project venster.
* Verwijder de GameManager nu uit de open scene en bewaar de scene.



* Open de StartScene, sleep de GameManager prefab naar deze scene en bewaar de scene.

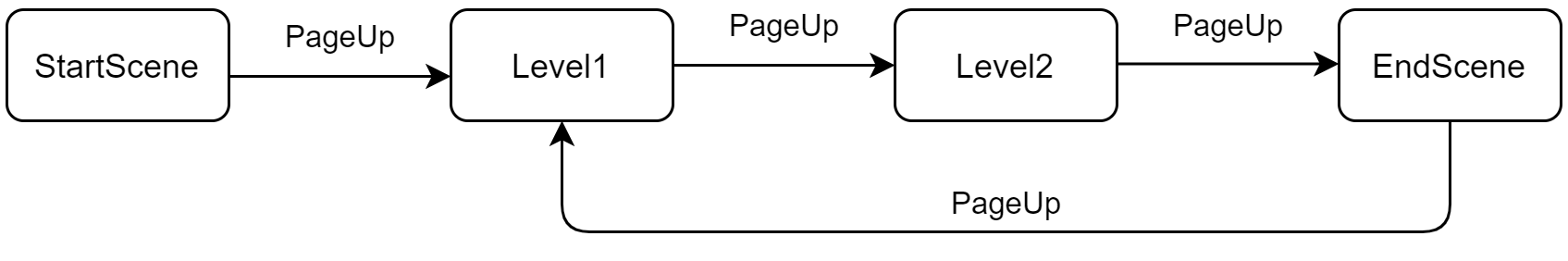


De StartScene moet ook aan de build settings worden toegevoegd.

* Open de Build Settings
* Sleep de StartScene naar de bovenste positie van de lijst met Scenes in de Build Settings.

### StartScene vermijden

We moeten nu voorkomen dat we ooit weer terug kunnen komen in de StartScene. Hieronder zie je schematisch wat de flow is.



* Pas de code zo aan, dat dit gerealiseerd wordt.

# Uitbreidingen

## Level2 herstarten

De bedoeling van *Level2* is dat de speler opnieuw moet beginnen als de bal van het speelveld af rolt.

Zorg ervoor dat *Level2* herstart als de bal van het veld rolt.

*Hint: een manier om te detecteren dat de bal van het speelveld is gerold is te controleren of de y-posititie van de bal onder een bepaalde waarde is gekomen.*

## Opbouw

In dit voorbeeld zit er een opbouw in de moeilijkheid van de levels. In het eerste level is er nog een muur zodat de speler veilig kan ontdekken hoe het spel werk en wat de bedoeling is. In het tweede level is deze veiligheid verwijderd.

Gebruik de resultaten van de eerdere Unity opdrachten om een spel met een aantal levels te maken waar een opbouw in zit.

## 