# Lesbrief 9: Unity

Een game is in Unity op te delen in stukken. Een game heeft meerdere **scenes**, één scene heeft meerdere **game-objecten**, een game-object heeft een of meerdere **componenten**. Dit kunnen voorgeprogrammeerde componenten van Unity zijn of jouw eigen scripts.

**Scenes**

|

|\_\_\_\_\_\_\_\_\_\_**Game-Objecten**

|

|\_\_\_\_\_\_\_\_\_\_**Componenten**

## Scène

| Levels in je game worden in Unity ‘scenes’ genoemd. Een scène wordt gevuld met allemaal game-objecten. Hiernaast zie je hoe alle game-objecten van een scène terug te vinden zijn in de ‘Hierarchy’.  De **Roll-a-ball** scène van de Unity-0 module heeft de volgende objecten:   * Player * Pick Ups * Directional Light * Camera   ...  In het volgende stukje gaan we het game-object van de speler bekijken als voorbeeld. |  |
| --- | --- |

## Game-objecten

Game-objecten bestaan uit componenten. Dit zijn stukjes logica die eigenschappen of gedrag van het game-object beïnvloeden.

Unity heeft al veel van deze componenten voor je geprogrammeerd. Deze gebruik je al. Links hieronder wordt het speler game-object met zijn componenten schematisch weergegeven. De componenten met een **Unity-logo** zijn al kant en klaar.

Maar het component genaamd **Player Controller** moet je zelf maken. Het volgende stukje legt dit uit.

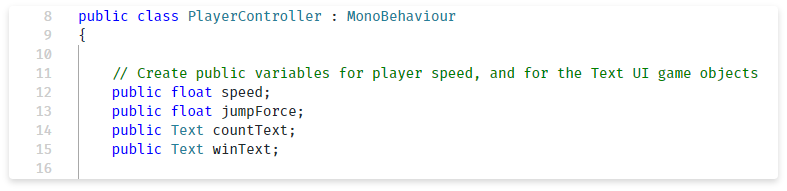
|  |  |
| --- | --- |

## Jouw eigen scripts

Als je in Unity een Script maakt, dan wordt er een **Class** gecreëerd. De Player Controller is dus een **klasse**.

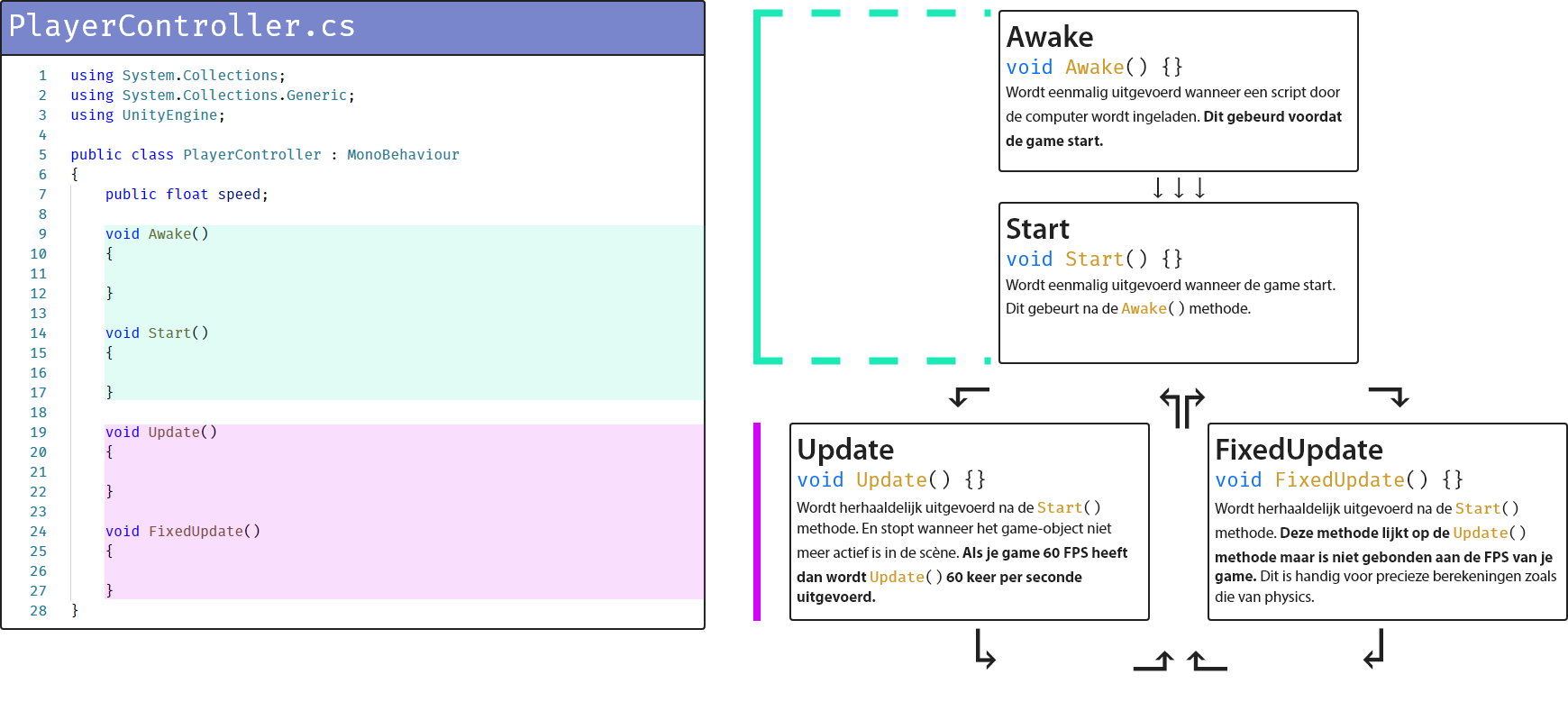
Vervolgens definieer je **variabelen** die nodig zijn voor je spel.

| Als je in jouw scripts variabelen public maakt dan kun je in Unity deze variabelen aanpassen en een waarde geven. Hiernaast zie je hoe de variabelen uit het script hieronder te zien zijn in de Unity editor. |  |
| --- | --- |



### Game Loop: de methoden

Jouw eigen script wordt door 4 methods aangestuurd. Deze worden in een volgorde en herhaaldelijk uitgevoerd. Door jouw eigen code hier in te schrijven kun je het game object aansturen. Hieronder staat links een leeg script met de 4 methods: *Awake()*, *Start()*, *Update()*, *FixedUpdate()*.



#### Awake methode

De awake methode wordt **eenmalig** uitgevoerd voordat de game of scene start. De awake methode is bedoeld om globale variabelen, variabelen die voor je hele spel gelden, of game states te initialiseren.

Met andere woorden als je spel al is gestart en je maakt nieuwe objecten aan dan wordt alleen de start methode uitgevoerd en niet de awake. Want de awake methode wordt alleen maar uitgevoerd als de game of scene nog niet is gestart.

#### Start methode

De Start methode wordt **eenmalig** uitgevoerd door het script, voordat de overige methoden van het script worden aangeroepen. In de Start methode initialiseer je variabelen van je script. Als de Start methode van je script wordt aangeroepen is er reeds een Game Object aangemaakt. Oftewel bij de initialisatie van het Game Object wordt de start methode aangeroepen.

#### Update methode

De Update methode wordt **herhaaldelijk,** per frame, uitgevoerd door je spel. Deze methode is afhankelijk van het aantal **fps** (frames per second) van je spel. Dit betekent dat als je spel 60 fps heeft dat de Update methode 60 keer per seconde wordt uitgevoerd. In de Update methode kun je bijvoorbeeld checken of de speler bepaalde toetsen indrukt.

#### FixedUpdate methode

De Fixed Update methode wordt **herhaaldelijk** uitgevoerd **maar is niet afhankelijk van het aantal fps**. De Fixed Update methode wordt aangeroepen door het Physics systeem.

De Fixed Update methode wordt gemiddeld vaker uitgevoerd dan de Update methode en kan voor of na een Update methode worden aangeroepen.

In de Fixed Update methode kun je bijvoorbeeld de verplaatsing van game objecten programmeren.