Jonathan Vercammen

Voorstudie

Framework

Er zijn drie verschillende frameworks om een mobiele app te schrijven: native, hybrid en PWA. Native is de beste wanneer je wil gaan voor maximale performantie en functionaliteit.

Hybrid apps zijn iets makkelijker om te maken maar zijn meer beperkt in functionaliteit. Frameworks zoals React Native en Electron gebruiken beide JavaScript als hun hoofdtaal. Dit maakt het eenvoudig om te leren.

Progressive Web Apps, of PWA's, maken gebruik van webframeworks met veel tools om je te helpen. Dat is als je ze kent tenminste. Vue of React zijn beide krachtige frameworks met de mogelijkheid om code te hergebruiken in components maar zijn niet altijd beginnervriendelijk.

Aangezien ik een spel maak, lijkt het mij logisch om een game-engine te gebruiken. De twee bekendste engines zijn Unity en Unreal Engine.

Unity is heel bekend doordat het een kleine instapcurve heeft en een heel grote community. Er zijn ook tal van assets die je kan gebruiken in je project zowel betalend als gratis.

Unreal Engine (UE) is de top van game-engines, zeker op grafisch vlak. Alleen is het veel moeilijker om te leren met al de functionalteit die UE biedt.

Beide engines kunnen zowel android als iOS apps maken maar ik kies ervoor om met Unity te werken omdat ik er al mee gewerkt heb.

Ik zou Unity classificeren als een hybrid framework omdat je voor meerdere platformen kan schrijven. Het is zeker geen native of web app.

Terrein maken

Kan ofwel met sprites of met een tilemap. Sprites geven meer controle over de vorm van het terrein. Een tilemap heeft meer structuur en werkt veel sneller. Voor een simpele 2D-platformer volstaat een tilemap. In Unity kan ik een grid maken met variabele grootte. Elke punt kan ik een bepaalde waarde geven bv. 0 voor lucht en 1 voor steen.

Touch-input van GSM verwerken

Voor mijn game heb ik touch-input nodig voor zowel menu's en om het personage te laten rondlopen. Daarvoor moet ik knoppen op het scherm plaatsen die de gebruiker kan indrukken. Unity heeft hier twee opties voor.

De eerste is de ingebouwde Input-klasse. Met deze manier moet je voor elke actie, bv. lopen of springen, zelf een toets instellen voor elk inputtoestel die onveranderlijk is. Dit is op zich niet zo'n groot probleem als je enkel voor Android schrijft.

De tweede is de nieuwe InputSystem 2.0. Dit is een installeerbare package die alle vormen van inputs probeert samen te brengen zodat je niet voor elk toestel aparte code moet schrijven. Voor mijn gebruik volstaan onderstaande klassen.

Touchscreen: Houdt alle touches bij.

TouchControl: Stelt een touch-input voor.

De grootste reden waarom ik de nieuwe InputSystem gebruik is events. Nu hoef je niet meer elke frame te controleren of een knop is ingedrukt, ingedrukt blijft of losgelaten wordt. Nu kan je hiervoor naar events luisteren. Dit bespaart veel debugwerk en hoofdpijn.

Met callbackmethodes in de InputAction-klasse weet ik wanneer een nieuwe touch wordt gemaakt of verwijderd.

Sensor-input

Het grootste probleem met de oude Input is het gebrek aan sensorinput. De standaardsensoren zoals de gyroscoop of de acceleratiesensor zijn er maar de licht- of nabeiheidssensor niet. Je moet ook zelf nog rekening houden met de zwaartekracht wanneer je de acceleratiesensor uitleest.

Dus ook hier maak ik gebruik van Unity's nieuwe InputSystem om alle sensoren van je GSM op te vragen en uit te lezen. Om een sensor aan te spreken, moet hij eerst geactiveerd worden bv.
InputSystem.EnableDevice(Gyroscope.current)

De Accelerometer-klasse in Unity meet ook de versnelling van de zwaartekracht. Dit maakt het lastig als je altijd rekening moet houden met de richting van je GSM om de zwaartekracht er uit te filteren.

Daarom zijn er twee extra klassen namelijk LinearAccelerationSensor, die enkel de relatieve versnelling meet en GravitySensor, die de absolute versnelling meet inclusief zwaartekracht.

Verder gebruik ik ook de AttitudeSensor (oriëntatie van GSM), ProximitySensor en LightSensor.

Ik heb al een kleine demo in Unity gemaakt om het uitlezen van sensoren te testen. Na uren debuggen en opzoeken blijkt dat de sensoren enkel werken als je het project build en runt op je GSM zelf.

Met deze informatie weet ik hopelijk alles om een werkende game te maken. Het moeilijkste zal zijn om de puzzels te bedenken en te maken. Dan moet ik zorgen dat ik de sensoren goed uitlees en niet in de war geraak met de drie verschillende assen (x,y,z).

Referenties

https://docs.unity3d.com/Packages/com.unity.inputsystem@1.0/manual/Sensors.html

How to use Unity's Input System

How to use NEW Input System Package! (Unity Tutorial - Keyboard, Mouse, Touch, Gamepad)

How to use Touch with NEW Input System - Unity Tutorial

TOUCH CONTROLS in Unity!

Tilemap in Unity (Build Worlds Easily)