Multiplatform development

Aaron Oostdijk

Vincent Creveld
3019866

GitHub link:
https://github.com/VincentCreveld/CreveldVincent_MPDEV

Uitleg game concept

Voor mijn multiplatform development project heb ik besloten om een endless runner-achtig spel te maken. De bedoeling is dat je als de auto stuurt om de dozen op de weg te vermijden en zo een zo hoog mogelijke score te behalen.

Op de PC kan je met A/D of de links en rechts pijltoetsen de auto naar links en rechts sturen en je kan met de spatiebalk je boost gebruiken.

Op Android kan je de auto besturen door het apparaat te draaien alsof je een echt stuur vasthoud (de x-as). De boost kan je gebruiken door de UI knop in te drukken.

Platform specifieke oplossingen

Op een Android apparaat heb je geen toegang tot een toetsenbord, dus heb ik de boost functionaliteit op een UI knop gezet. Het sturen gebeurt ook met de gyroscoop in de telefoon.

Code opzet

De input binnen het project wordt heel simpel gehanteerd; de PlayerManager vraagt tijdens de initialisatie welk platform het op gespeeld wordt, en voegt de stuur- en boostfunctie toe aan de bijbehorende events binnen de platform-specific InputManager.

De PlayerManager heeft de Boost() en Move() functie omdat de movement niet anders hoort te zijn per platform, maar de input wel. Dit is dus waarom ik de input als events doorgeef tussen de PlayerManager en de InputManagers.

De platform-specifieke InputManager wordt door middel van een Abstract Factory aan de player teruggegeven.

```
//Get controls form Abstract factory
private ControlSet currentControls;
```

```
currentControls = ControlSet.GetControls();
currentControls.Start();
currentControls.moveInput += Move;
currentControls.jumpInput += Boost;
```

Initialisatie van de controls in de PlayerManager.

```
public delegate void ControlEventJump();
public delegate void ControlEventAxis(float axis);

public class ControlSet {

    public ControlEventJump jumpInput;
    public ControlEventAxis moveInput;

    public virtual void Start() {
    }

    public static ControlSet GetControls() {
        if(Application.isEditor)
            return new PcControls();
        switch(Application.platform) {
            #if !DISABLE_SYSTEM
            case RuntimePlatform.WindowsPlayer:
                return new PcControls();
            case RuntimePlatform.Android:
                return new AndroidControls();
            #endif
            default:
                return new DummyControls();
        }
    }

    public virtual void CheckInput() {
    }

}
```

Abstract factory + events in de parent-class. De editor heeft dezelfde implementatie als de PC build gekregen.

```
public class DummyControls : ControlSet {

    public override void Start() {
        base.Start();
        Debug.Log("No controls initialised, platform not compatible!");
        Application.Quit();
    }
}
```

Wanneer het spel op een non-supported platform opstart sluit het zichzelf gelijk af.

```
public class PcControls : ControlSet {

    public override void Start() {
        base.Start();
        Debug.Log("Pc controls initialised!");
    }

    public override void CheckInput() {
        base.CheckInput();

        //Gets A/D or <-/-> input
        moveInput(Input.GetAxisRaw("Horizontal"));

        //Gets jump input
        if(Input.GetButtonDown("Jump")) {
            jumpInput();
            Debug.Log("Boost!");
        }
    }


}
```

De PC controls zien er simpel uit omdat ik in de control-hiërarchie ervoor gekozen heb om alleen de controls te laten verwerken. Dit heb ik bereikt door middel van events.

```
public class AndroidControls : ControlSet {

    private RectTransform androidControls;

    public override void Start() {
        base.Start();
        Debug.Log("Android controls initialised!");
        Screen.orientation = ScreenOrientation.LandscapeLeft;
        androidControls = GameObject.Find("AndroidControls").GetComponent<RectTransform>();
        androidControls.GetChild(0).gameObject.SetActive(true);

        //Sets the UI button's function to the Boost function
        androidControls.GetChild(0).GetComponent<Button>().onClick.AddListener(Boost);
    }

    public override void CheckInput() {
        base.CheckInput();

        //Gets A/D or <-/-> input
        moveInput(Input.acceleration.x);
    }

    public void Boost() {
        jumpInput();
    }
}
```

In de android controls zoek ik en zet ik de UI button aan zodat een Android gebruiker ook kan boosten.