Crappy Plane

A Flappy Bird spin-off, but turn based

Alex van der Linden
GDEV2
3016859

Samenvatting

Voor de opdracht maak een turn-based multiplayer spel' wilde ik een bestaand spel proberen aan te passen aan de opdracht. Ik hou van spellen waar je als team iets moet voltooien, dus waarom geen turn-based Flappy Bird?

Crappy Plane is een spin-off van het spel *Flappy Bird*. In Flappy Bird is het de bedoeling dat de speler klikt om de vogel omhoog te laten 'springen' waarna deze weer naar beneden valt. Door obstakels te ontwijken kan de speler een score behalen.

In Crappy Plane besturen twee tot vier mensen om de beurt een vliegtuigje die rotsen moet ontwijken. Wanneer de speler klikt gaat het vliegtuigje omhoog die dan weer langzaam omlaag valt. Als een speler heeft geklikt, gaat de beurt naar de volgende speler in het spel. Dit resulteert in een spelervaring met een snel tempo waar spelers samen moeten werken om het doel (een hoge score) te bereiken.

Photon

Het spel gebruikt Photon Unity Networking (*PUN*). Ik vond PUN makkelijk in gebruik na wat minder goede experimentele ervaringen met Unity Networking (*UNET*).

PUN gebruikt in principe hetzelfde systeem als UNET, maar voelt gebruiksvriendelijker aan. Bijna alle gebeurtenissen hebben een callback waar je eigen functionaliteit aan toe kan voegen.

Flow

De speler komt binnen in het spel waar hij/zij een naam in kan vullen die anderen in het spel ook kunnen zien. De speler verbindt daarna met de PUN server (gebaseerd op locatie) en komt in een scherm waar het spel afgesloten kan worden of er naar door spelers gemaakte lobby's kan zoeken. Bij het zoeken van een lobby staat de naam van de lobby, hoeveel spelers er al inzitten en of deze met een wachtwoord beveiligd is. Na een lobby in te zijn gegaan, moeten alle spelers op 'ready' klikken waarna de host het spel kan starten. Er is ook een mogelijkheid om zelf een lobby te creëren. De speler kan dan een naam voor de lobby kiezen en deze starten.

In het spel wordt er een volgorde van spelers aangemaakt die op elke *client* hetzelfde is en kunnen spelers om de beurt klikken op het vliegtuigje in de lucht te houden.

Assets
Alle gebruikte assets zijn van *Kenney*, een Nederlandse ondernemer.

Bekende problemen
- Na het voltooien van een spel gaat de speler terug naar het hoofdmenu in plaats van de lobby
- Het het voltooien van een spel blijft de speler (de NIET host) soms vastzitten in het spel
    - Druk op de ESC knop om een menu te openen om alsnog het spel te verlaten
- De controller (het vliegtuigje) werkt niet optimaal. Als spelers te snel achter elkaar klikken gaat het vliegtuigje amper omhoog
    - Mogelijke oplossing: Vector2 NIET resetten als de omhoog waartse beweging groter is dan nul
- Spel kan gestart worden met maar één persoon in de lobby
- Er zit nog geen score systeem in het spel
    - Elk ontweken obstakel moet +1 geven
- Het spel start direct als de host op 'start' heeft geklikt
- Filter bij het zoeken van een lobby is nog niet geïmplementeerd
- Het is niet mogelijk om een wachtwoord voor een lobby aan te maken
    - Dit is een functie die niet beschikbaar is binnen PUN

Door te weinig tijd
- Het inloggen (opdracht Ton) is niet in het spel verwerkt, maar op een webpagina te vinden
- Ping van elke client weergeven nog niet geïmplementeerd

Belangrijkste werking

'Inloggen' (LoginMenu.cs)

Via PhotonNetwork kan functionaliteit worden aangeroepen voor de client.

Bij het opstarten van het spel:
- PhotonNetwork.playerName
    - Sla de naam van de speler op in PhotonPlayer
- PhotonNetwork.automaticallySyncScene
    - Zorgt dat elke client in de lobby dezelfde scene laadt als de host
- PhotonNetwork.ConnectUsingSettings (string versie)
    - Verbindt met PUN met de client versie

Lobby zoeken (RoomBrowserMenu.cs)

Er wordt een lijst met beschikbare lobby's getoond. Deze worden opgehaald met:

- PhotonNetwork.GetRoomList()
    - Geeft RoomInfo[] terug

De gegevens hiervan worden verwerkt in een visuele lijst met een 'Join' knop om de lobby in te gaan:

- PhotonNetwork.JoinRoom(string lobbyNaam)

Maken van een lobby (RoomCreationMenu.cs)

Een lobby kan worden aangemaakt met:

- PhotonNetwork.CreateRoom(string lobbyNaam, RoomOptions opties, TypedLobby lobbyType)

In de opties is er de mogelijkheid om via een Hashtable optionele opties toe te voegen (bijvoorbeeld welke map om in te spelen er gekozen is). Binnen de RoomOptions zijn er variabelen aan te passen waaronder het maximaal aantal spelers en de naam van de lobby.

Wachtruimte

Elke client in de lobby krijgt een callback wanneer er een speler zijn/haar eigen opties aanpast (in de PhotonPlayer), er iemand connect en disconnect.

- OnPhotonPlayerConnected(PhotonPlayer)
    - De visuele interface wordt geüpdate
- OnPhotonPlayerDisconnected(PhotonPlayer)
    - De visuele interface wordt geüpdate
- OnMasterClientSwitched(PhotonPlayer)
    - Een nieuwe host wordt gekozen en de visuele interface wordt geüpdate

Elke client heeft een 'Ready' knop ('Start' knop wanneer de client de host is). Wanneer hier op geklikt wordt, komt er een callback op elke client en worden de opties van de PhotonPlayer aangepast. Alleen als ALLE clients *ready* zijn kan de host het spel starten.