Rocket Boost: Un Gioco in Unity 6

Asia Mazzotta

1 Introduzione

Rocket Boost 'e un gioco sviluppato con Unity 6 nell'ambito del corso *Complete C# Unity Game Developer 3D* di Rick Davidson e del team di GameDev.tv su Udemy.

2 Gameplay

Il gioco è composto da due livelli, in cui il giocatore deve pilotare un razzo da una piattaforma all'altra evitando ostacoli. Il razzo può muoversi solo a destra e sinistra, ma rimane bloccato lungo l'asse Z.

In caso di collisione con un ostacolo o di vittoria (atterraggio corretto), vengono avviati due diversi sistemi di particelle per fornire un feedback visivo al giocatore.

3 Primo Livello

Il primo livello presenta un'ambientazione semplice ma funzionale, con elementi di sfondo che contribuiscono a creare un contesto visivo coinvolgente. L'elemento centrale è il razzo, posizionato tra due piattaforme che fungono da ostacoli e punti di atterraggio.

L'obiettivo principale di questo livello è acquisire familiarità con le tecniche e gli strumenti dell'editor di Unity, sperimentando l'implementazione di script per il controllo del razzo. In particolare, è stato utilizzato il metodo `AddRelativeForce` per applicare forze al velivolo, sfruttando il sistema fisico dell'engine. Il controllo del razzo avviene tramite `InputSystem`, che permette di rilevare e interpretare gli input dell'utente in modo efficace.

Oltre al movimento, è stata implementata la gestione delle collisioni mediante uno script dedicato, che rileva gli impatti con piattaforme e altri oggetti determinando le reazioni appropriate. Inoltre, è stato integrato un sistema per la chiusura dell'applicazione in determinate condizioni.

Per migliorare l'esperienza visiva, alcuni elementi dello sfondo sono stati animati, aggiungendo dinamicità all'ambientazione. Questo contribuisce a rendere il livello più immersivo, migliorando il senso di profondità e movimento all'interno della scena.

4 Secondo Livello

Il secondo livello mantiene la stessa ambientazione del primo, realizzata tramite prefab, ma introduce ostacoli in movimento che si spostano attraverso i metodi `PingPong` e `Lerp` del modulo `Mathf`.

Per arricchire ulteriormente l'esperienza di gioco, è stato aggiunto un `AudioSource` al razzo e sono stati utilizzati effetti sonori forniti tra gli asset del corso. Gli effetti audio vengono riprodotti quando il razzo si scontra, raggiunge la piattaforma o si muove, migliorando il coinvolgimento del giocatore.

5 Implementazione

Per gestire il comportamento del razzo sono stati creati due script, entrambi applicati allo stesso oggetto:

5.1 PlayerMovement

Si occupa di registrare gli input della tastiera relativi al movimento e alla spinta del razzo. Una volta registrata l'azione del giocatore, applica la forza corrispondente e gestisce la produzione dei suoni e degli effetti visivi.

Listing 1: Codice per applicare la spinta

```
void ProcessThrust()
{
    if (thrust.IsPressed())
    {
        rb.AddRelativeForce(Vector3.up * thrustStrength * Time.fixedDeltaTime);
        if (!audioSource.isPlaying)
        {
            audioSource.PlayOneShot(mainEngineSFX);
        }
        if (!thrustParticle.isPlaying)
        {
            thrustParticle.Play();
        }
    }
    else
    {
        StopThrusting();
    }
}

void StopThrusting()
{
    thrustParticle.Stop();
```

```
    audioSource.Stop();
}
```

5.2 Collision

Si occupa di controllare, in caso di collisione del razzo con altri gameobject, il tipo di quest'ultimo e gestisce la logica del gioco, come ad esempio il reset del livello in caso di collisione con un ostacolo o il passaggio al livello successivo in caso di completamento.

Listing 2: Esempio di gestione del reset del livello o passaggio a quello successivo

```
void LoadNextLevel()
{
    int currentScene = SceneManager.GetActiveScene().buildIndex;
    int nextScene = currentScene + 1;
    if (nextScene == SceneManager.sceneCountInBuildSettings)
    {
        nextScene = 0;
    }
    SceneManager.LoadScene(nextScene);
}

void ReloadLevel()
{
    int currentScene = SceneManager.GetActiveScene().buildIndex;
    SceneManager.LoadScene(currentScene);
}
```

6 Evoluzione futura

L'idea per il futuro è di espandere il gioco aggiungendo nuovi livelli con ostacoli di comportamento differente e livelli con difficoltà crescente. Questo permetterà di rendere l'esperienza di gioco pi'u varia e sfidante, introducendo meccaniche avanzate e nuovi elementi interattivi.

7 Conclusione

Rocket Boost è stato sviluppato per apprendere le basi della creazione di giochi in Unity. Il progetto ha permesso di esplorare l'editor, comprendere i principali strumenti di sviluppo e gestire dinamiche di gioco come fisica, animazioni e ostacoli interattivi. Questo rappresenta un primo passo verso la progettazione di esperienze di gioco più complesse e strutturate.