Wydział Elektroniki i Technik Informacyjnych

Programowanie gier komputerowych

Projekt

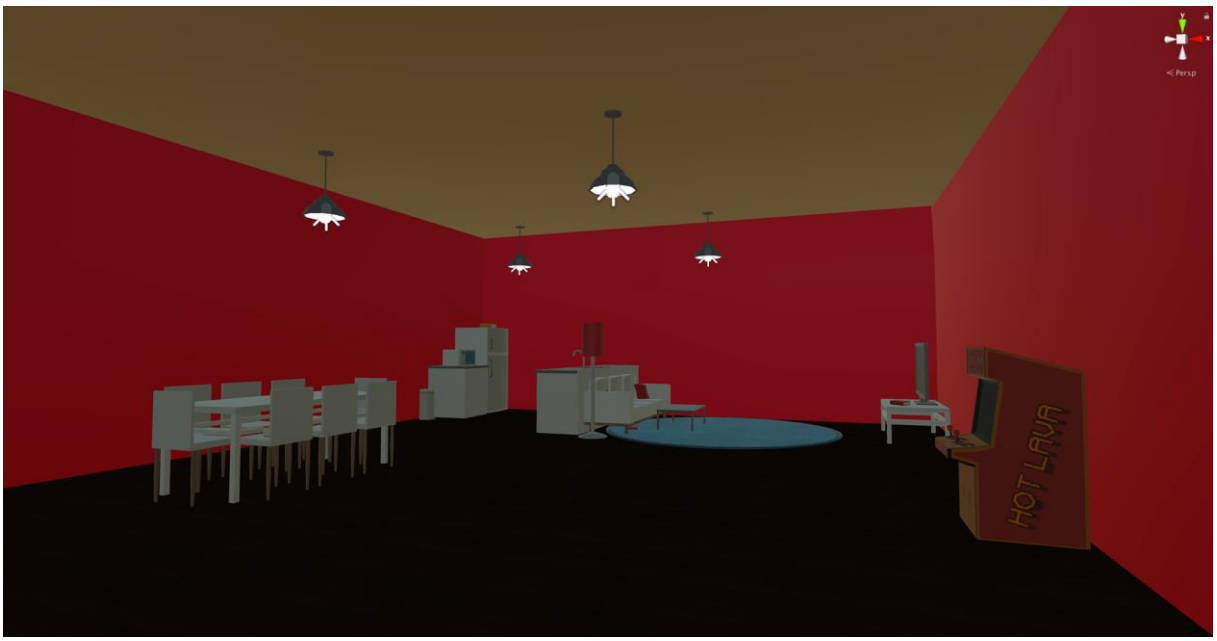
Mikołaj Płachta, 250912
Mariusz Pajączkowski, 242714

1. Opis gry

Zaprojektowano i wykonano trójwymiarową grę logiczno-przygodową, w której gracz prowadzi bohatera w widoku z pierwszej osoby w jego zmaganiach z zaprojektowanymi przeszkodami. Rozgrywka została podzielona na trzy rozdziały wymagające od użytkownika użycia różnych umiejętności.

Pierwszy poziom to część logiczna. W labiryncie znajdującym się w mrocznym lesie należy odnaleźć prawidłową ścieżkę. Na końcu każdej z nich czekają na gracza drzwi, z których tylko jedna prowadzi na drugi poziom. W celu ułatwienia poszukiwań przy pozycji startowej umieszczono podpowiedź w formie tabliczki, w której brakuje liter „L" oraz „P". Są to pierwsze litery kierunków z języka angielskiego. W przypadku znalezienia błędnej ścieżki drzwi przenoszą postać na pozycję startową.

Drugi poziom przenosi gracza do opuszczonego domku, w którym otrzymuje zadanie zebrania poukrywanych książek.



Po zebraniu wszystkich 10 przedmiotów, uruchamiana jest możliwość użycia automatu do gier. Trudność stanowi fakt, że gracz nie ma pojęcia o ilości książek, które należy zebrać. Dopiero po zebraniu wszystkich zostaje on poinformowany stosownym napisem.

Trzeci poziom jest uruchamiany po użyciu automatu do gier. Przenosi on gracza do gry zręcznościowej (gra w grze), w której postać ma zadanie przejść do końca planszy. Plansza dzieli się na kafelki bezpieczne i niebezpieczne. Te drugie odbierają bohaterowi życie, które jest wyrażone w formie liczbowej. W przypadku spadku życia do 0 postać umiera i gracz wraca do poziomu drugiego, gdzie po interakcji z automatem może ponownie rozpocząć poziom trzeci. Po udanym pokonaniu niebezpieczeństw postać natrafia na drzwi, które prowadzą do miejsca zakończenia gry.

2. Technologia

Grę wykonano za pomocą oprogramowania Unity3D stanowiące niezwykle popularne narzędzie w dzisiejszych czasach. Największą zaletą programu jest możliwość łatwego pisania skryptów w wysokopoziomowych językach jak C# czy JavaScript. Do realizacji wydarzeń i logiki użyto języka C# ze względu na dobrą znajomość tej technologii.

```
// Update is called once per frame
void Update ()
{
    if(CheckPlayerInDoors(player.transform.position, this.transform.position)) {
        if ((Mathf.Round(this.transform.position.x) == 124) && (Mathf.Round(this.transform.position.z) == 221)) {
            player.transform.position = new Vector3 (230.0f, 1.0f, 95.0f);
            endTime = Time.time + 5;
        } else if ((Mathf.Round(this.transform.position.x) == 383) && (Mathf.Round(this.transform.position.z) == 109)) {
            player.transform.position = new Vector3 (314.0f, 1.3f, 164.0f);
            renderWin = true;
        } else {
            player.transform.position = new Vector3 (35.0f, 0.0f, 214.0f);
        }
        renderBooks = false;
    }
}

bool CheckPlayerInDoors(Vector3 playerPos, Vector3 doorPos)
{
    if((Mathf.Abs(doorPos.x - playerPos.x) <= offset) && (Mathf.Abs(doorPos.z - playerPos.z) <= offset))
    {
        return true;
    }
    return false;
}

void OnGUI()
{
    if (renderBooks)
    {
        GUI.Label (new Rect (Screen.width * 0.5f - 200.0f, Screen.height * 0.5f - 10f, 10, 20), "Zbierz wszystkie książki. Użyj E.", style);
    }
    if (renderWin)
    {
        GUI.Label (new Rect (Screen.width * 0.5f - 200.0f, Screen.height * 0.5f - 10f, 10, 20), "Gratulacje! Wygrałeś! Koniec gry :)", style);
    }
}
```

```
public Transform transform { get; }

Summary
The Transform attached to this GameObject.
```

Dodatkowo wykorzystano oprogramowanie Adobe Flash Professional do tworzenia tekstur.

3. Ciekawe rozwiązania techniczne

W pierwszym poziomie do otoczenia ścieżek użyto specjalnie zaprojektowanych modeli drzew wygenerowanych przez narzędzie dostępne w pakiecie Unity. Za jego pomocą można tworzyć drzewa z jednym lub wieloma konarami głównymi, tworzyć konary pośrednie, ustawiać ich rozstawienie, nachylenie, ilość czy nawet szanse na złamanie gałęzi i miejsce tego złamania. Na koniec dodano liście na podstawie gotowej tekstury i ustawiono ich pozycję na wcześniej przygotowanych gałęziach. Dodatkowo ustawiono obiekt wiatru, który symuluje kołysanie się konarów i liści. Obsługę drzwi zrealizowano za pomocą skryptu, który sprawdza pozycję gracza i porównuje z obrysem wokół wszystkich portali. W przypadku odległości mniejszej niż ustalona aktywowana jest część kodu odpowiedzialna za teleportację postaci.

W drugim poziomie wykorzystano gotowe assety pozyskane z AssetStore do wykonania otoczki graficznej. Do każdej z szukanych przez postać książek dołączono skrypt, który po-

dobnie jak ten z poprzedniego etapu sprawdza odległość gracza od obiektu. Dodatkowo dopisano fragment wyzwalający zebranie książki poprzez naciśnięcie odpowiedniego przycisku. Po zebraniu obiektu zostaje on zniszczony, a globalna zmienna odpowiadająca za liczbę zebranych książek zostaje inkrementowana. W skryptach umieszczono także fragment, który odpowiada za wyświetlanie przed postacią wybranych tekstów i kontrolowanie procesu ich znikania w wybranym przez programistów czasie.

W trzecim poziomie zaimplementowano logikę do wyliczania aktualnego poziomu życia postaci i jego wyświetlania. Dodatkowo napisano dwa skrypty, które pozwalają na utrudnić rozgrywkę. Do dwóch ścian przypisano skrypt, który porusza je z zadaną prędkością wzdłuż poziomu oraz ustawia je w pozycji początkowej przy każdym uruchomieniu poziomu. Natomiast do czerwonych kafelków podłogi przypisano skrypt, który odbiera postaci życie w czasie styku bohatera z obiektami.

4. Dostępność gry

Niniejszą dokumentację wydrukowano i przekazano prowadzącemu oraz umieszczono w zdalnym repozytorium.

Pliki projektowe gry ze względu na ogromny rozmiar umieszczono w zdalnym repozytorium pod adresem: https://github.com/mxndev/UnityPGK/