Nowoczesne technologie programistyczne

Ćwiczenie

Temat: Pisanie skryptów w języku C# do silnika Unity

Michał Maciej L3

Contents

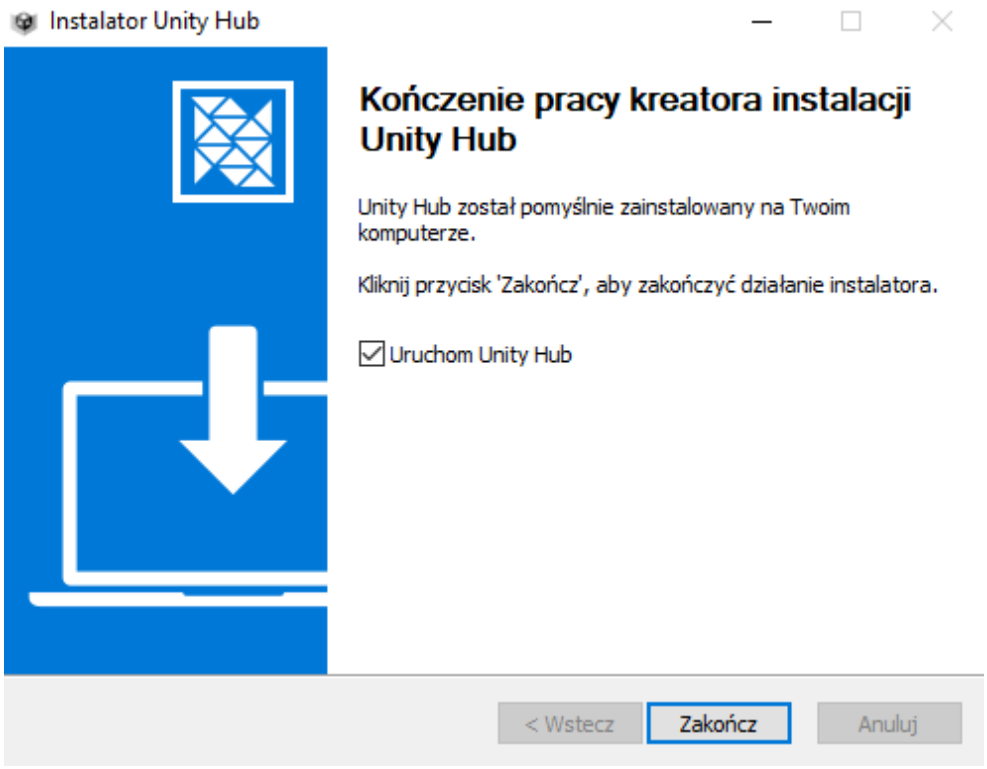
Cel ćwiczenia

Celem ćwiczenia jest zapoznanie się z interfejsem oraz pisaniem skryptów do silnika gier Unity.

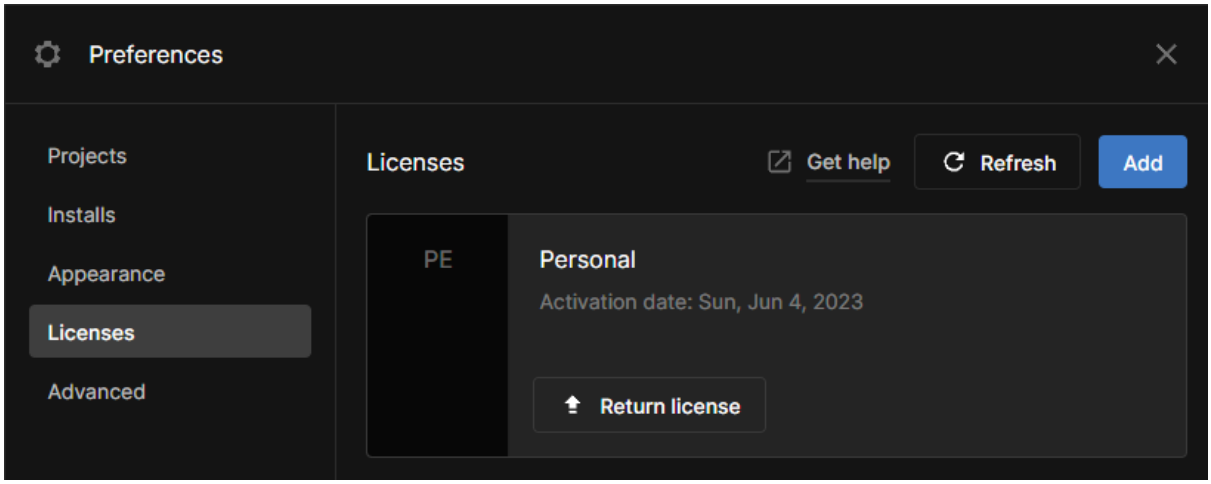
Instalacja środowiska

Unity

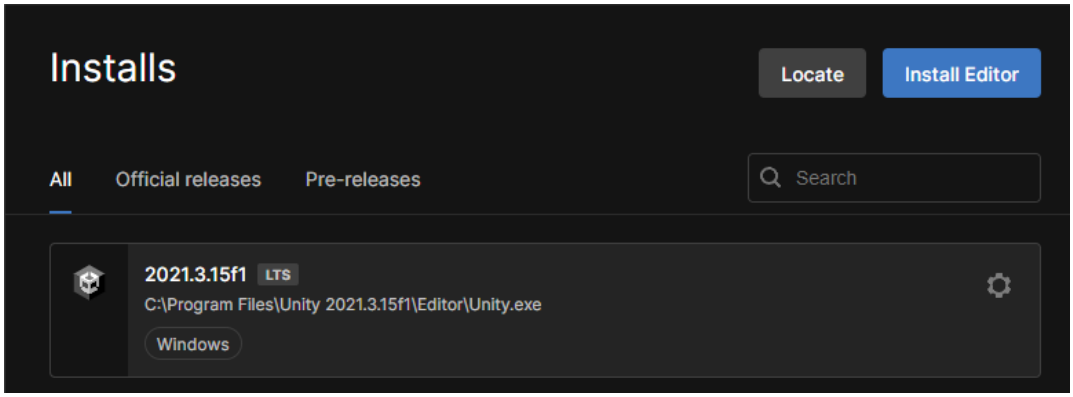
W celu zainstalowania silnika Unity pobrany i zainstalowany został Unity Hub.



Następnie Unity Hub został uruchomiony, utworzone zostało nowe konto oraz uzyskana została licencja wymagana do rozpoczęcia pracy.

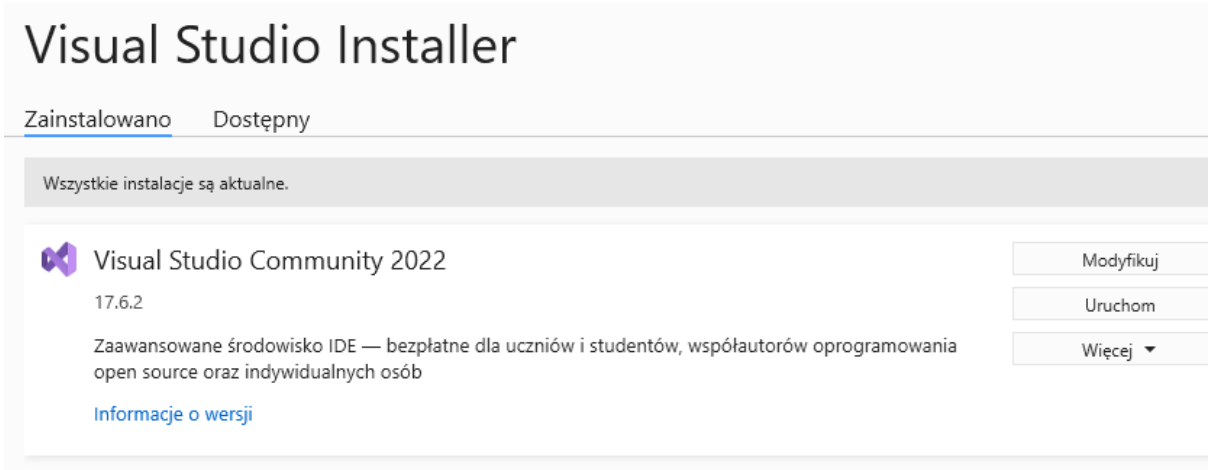


Wersja Unity 2021.3.15f1 została pobrana i dodana do Unity Hub.



Visual Studio

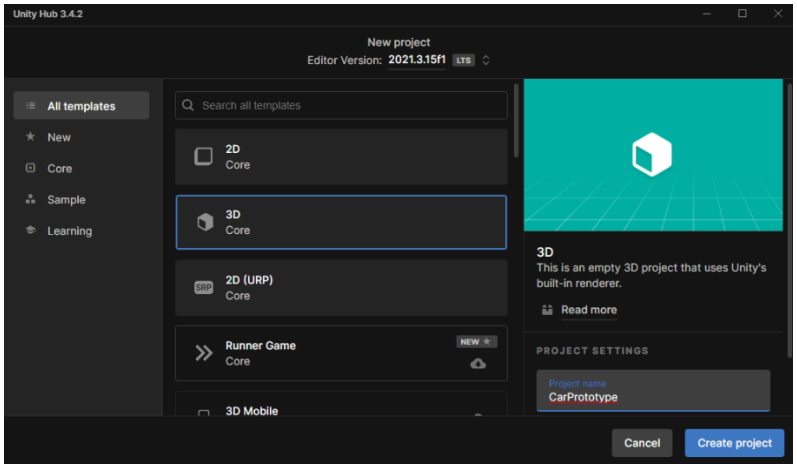
Pobrany został instaler Visual Studio w wersji Community. Przy instalacji zaznaczona została opcja Game development with Unity i środowisko zostało zainstalowane.



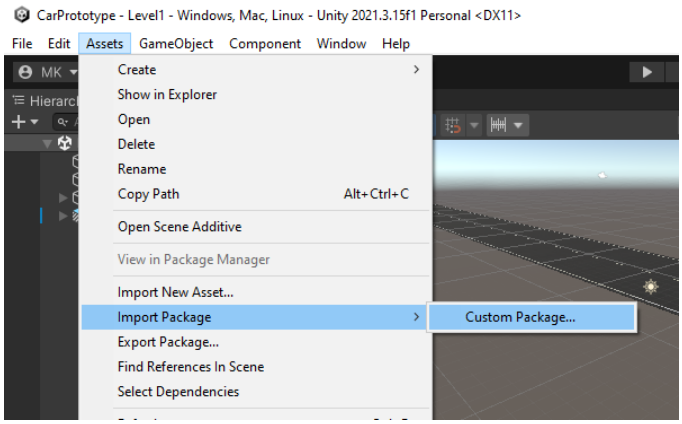
Następnie Unity zostało zintergrowane z Visual Studio.

Paczka z projektem

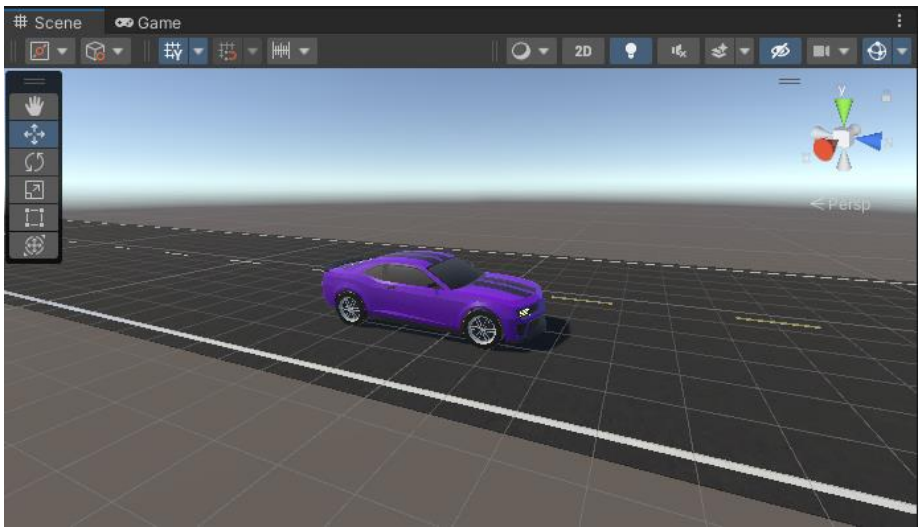
Paczka z projektem została pobrana z grupy na Teams, nastepnie utworzony w Unity został nowy projekt 3D „CarPrototype”.



Zaimportowanie paczki.



Efekt po wykonaniu wszystkich kroków.



Interfejs Unity

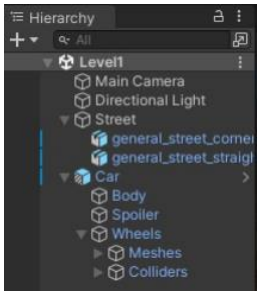
Zapoznałem się z najczęściej wykorzystywanymi oknami edytora takimi jak:

Pasek narzędzi

Najważniejszym elementem paska są elementy sterujące trybem odtwarzania

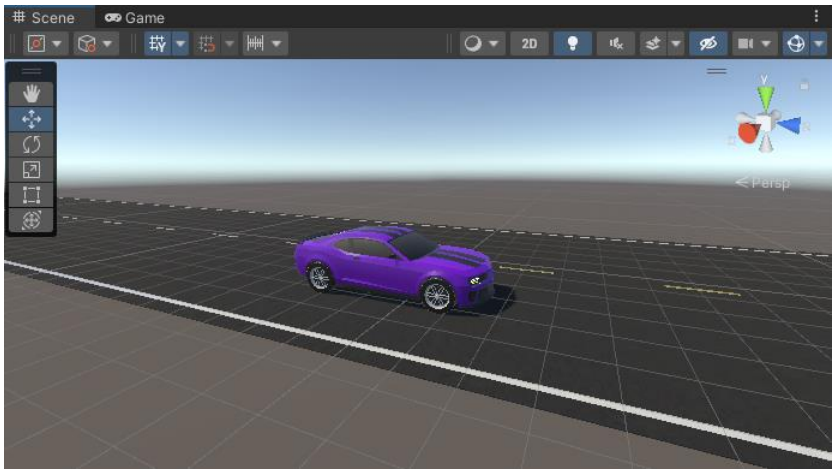
Okno hierarchii

W oknie hierarchii znajdują się wszystkie obiekty zawarte na scenie



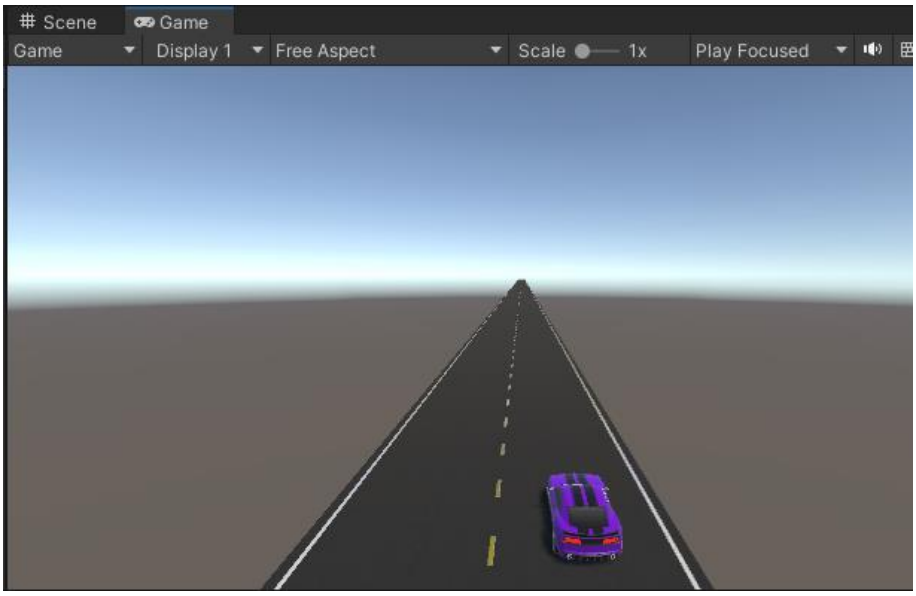
Okno sceny

Przedstawia widok sceny, tj. umieszczone na niej obiekty. Za pomocą tego okna mozna dodawać obiekty od sceny, zmieniać ich położenie, rotację itp.



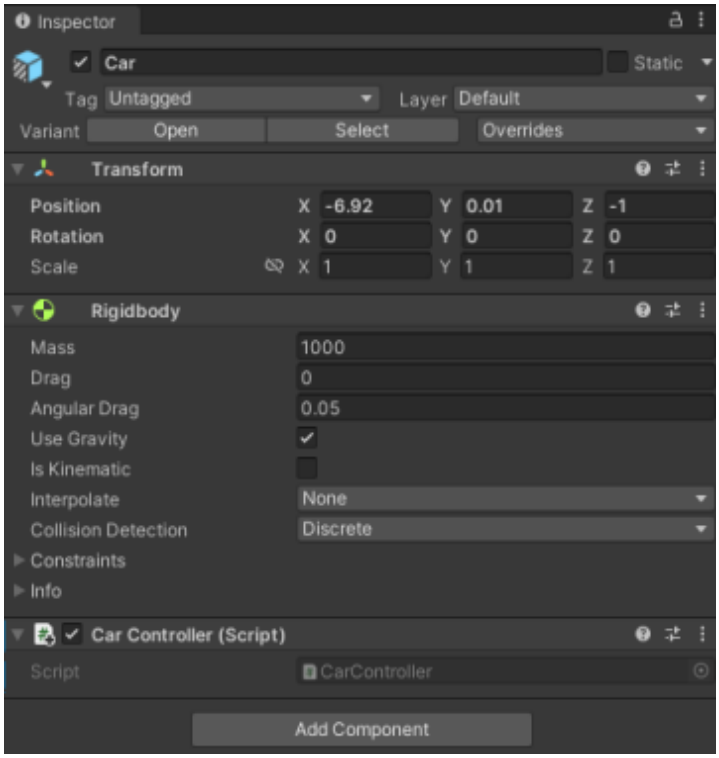
Okno gry

Przedtstawia widok gry, czyli to, co faktycznie jest widoczne po uruchomieniu gry z perspektywy gracza.



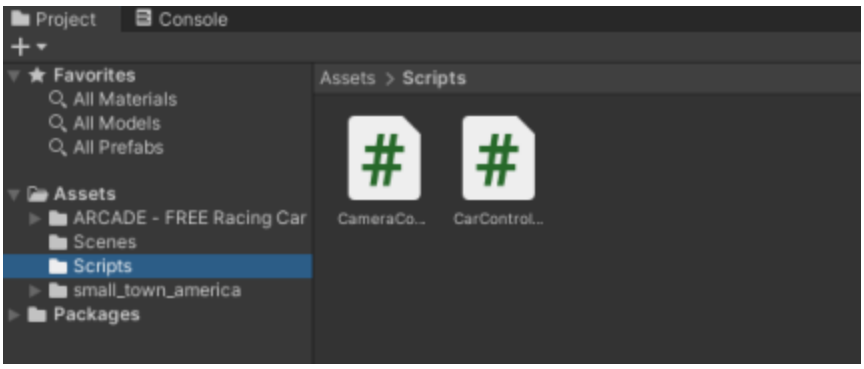
Inspektor

Umożliwia przeglądanie i dodawanie komponentów, obiektów (właściwości).



Okno projektu

Przedstawia wszystkie Assety (zasoby) projektu. Znajdują się tam skrypty, obiekty, sceny, tekstury itp.

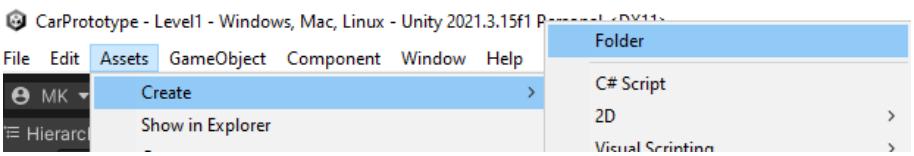


Podstawy programowania w Unity

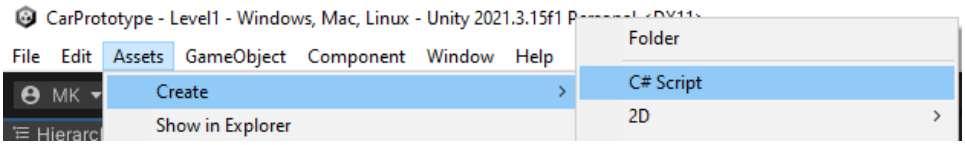
Pisanie skryptów do silnika Unity polega na utworzeniu pliku języka C# i dołączeniu go jako komponent odpowiedniego obiektu gry, którym ma sterować.

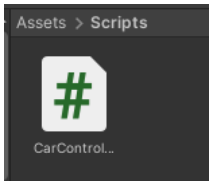
Dodanie skryptu do obiektu

Utworzenie podfolderu w folderze „Assets”.

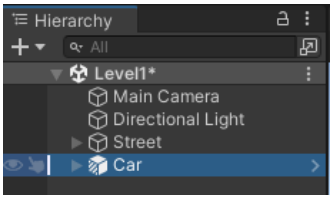


Utworzenie nowego skryptu C# „CarController”.

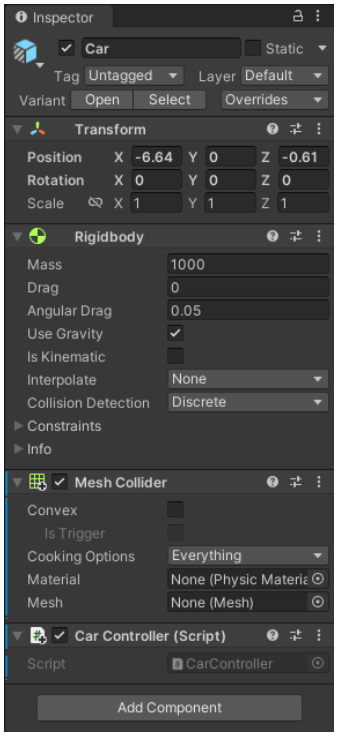




Dodanie utworzonego skryptu do wybranego obiektu czyli samochodu. W tym celu klikamy na obiekt w oknie Hierarchy.



Tak aby pojawiło się jego okno Inspector i jako komponent dodajemy wybrany nowo utworzony plik.



Pisanie skryptu sterującego obiektem

Wygląd nowo utworzonego skryptu.

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

Skrypt aparatu Unity | Odwołania: 0
public class CarController : MonoBehaviour
{
    // Start is called before the first frame update
    Unity Message | Odwołania: 0
    void Start()
    {

    }

    // Update is called once per frame
    Unity Message | Odwołania: 0
    void Update()
    {

    }
}
```

Metoda Start() – jest wywoływana jeszcze przed uruchomieniem gry, więc jest dobrym miejscem do wszelkich inicjalizacji.

Metoda Update() – jest wywoływana raz na klatkę, czyli cały czas w ciągu gry. To tutaj powinno się znaleźć sterowanie obiektem, gdyż będziemy ciągle sprawdzać, czy nie został naciśnięty klawisz odpowiadający za jakiś ruch.

Układ współrzędnych

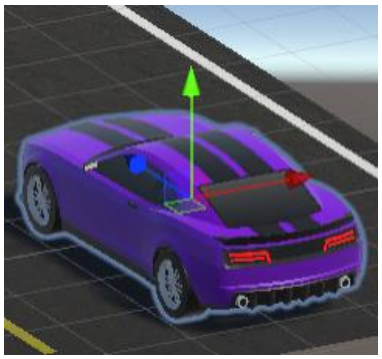
Jako iż pracujemy z grą 3D, to układ współrzędnych składa się z trzech osi: X, Y oraz Z.

Kierunki osi:

- oś X

- oś Y

- oś Z



Zmiana położenia obiektu

Za wszystkie zmiany położenia, rotacji i skali obiektów odpowiada komponent Transform, który posiada każdy obiekt

Aby odwoływać się do tego komponentu potrzebna będzie jego metoda Translate.

transform.Translate – zmienia położenie obiektu o daną wartość i w danym kierunku

Istnieje kilka sposobów użycia powyższej metody. Przykładowa jej struktura wygląda następująco:

public void Translate(float x, float y, float z);

Parametry x, y, z odpowiadają przesunięciu obiektu w kierunku danej współrzędnej.

Tak więc, by nasz obiekt zaczął się poruszać przed siebie, musimy wpisać jakąś dodatnią wartość liczbową typu float w miejsce argumentu Z. Moze być to przykładowo wartość 1.

transform.Translate(0.0f, 0.0f, 1.0f);

W ten sposób, wybrany obiekt będzie przesunięty o 0 jednostek względem osi X i Y oraz o 1 jednostkę względem osi Z (jedna jednostka powinna odpowiadac jednemu metrowi w grze).

Po zamieszczeniu powyższej linijki w metodzie Update(), samochód zaczął przemieszczać się do przodu.

Przydatne metody i atrybuty

Atrybuty klasy GameObject:

• **transform** - transformacja dołączona do tego obiektu.

Metody klasy GameObject:

• **Find** - znajduje obiekt GameObject według nazwy i zwraca go.

public static GameObject Find(string name);

Atrybuty komponentu transform:

• position - pozycja transformacji w przestrzeni świata.

• rotation - kwaternion przechowujący obrót transformacji w przestrzeni świata.

Metody komponentu transform:

• Translate - przesuwa transformacj¸e w kierunku i odległosci przesuni¸ecia. ´

public void Translate(float x, float y, float z);

• Rotate - wykonuje obrót obiektu gry.

public void Rotate(float xAngle, float yAngle, float zAngle);

Zadania

Zadanie 1

Zadanie numer 1 polegało na przekształceniu skryptu tak aby samochód jechał dwa razy szybciej niż oryginalnie. Szybkość samochodu zależy od tego o jaką wartość przesuwamy obiekt co klatkę w Update(), więc aby samochód poruszał się dwa razy szybciej musimy zamienić 1.0f na 2.0f.

Kod rozwiązania:

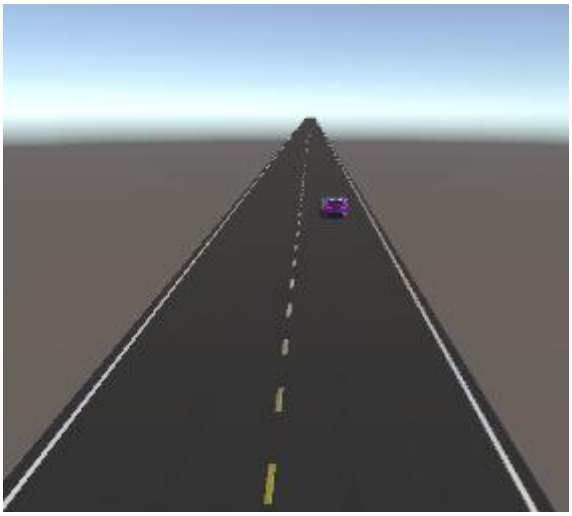
```
transform.Translate(0.0f, 0.0f, 2.0f);
```

Zadanie 2

Zadanie numer 2 polegało na przekształceniu skryptu tak aby samochód jechał cztery razy wolniej niż oryginalnie. Szybkość samochodu zależy od tego o jaką wartość przesuwamy obiekt co klatkę w Update(), więc aby samochód poruszał się cztery razy wolniej musimy zamienić 1.0f na 0.25f.

Kod rozwiązania:

```
transform.Translate(0.0f, 0.0f, 0.25f);
```



Zadanie 3

Zadanie numer 3 polegało na przekształceniu skryptu tak aby samochód jechał w przeciwną stronę niż oryginalnie. Kierunek jazdy samochodu zależy od tego o jaką wartość przesuwamy obiekt co klatkę w Update(), więc aby samochód poruszał się w tył musimy zamienić 1.0f na -1.0f.

Kod rozwiązania:

```
transform.Translate(0.0f, 0.0f, -1.0f);
```

Zadanie 4

Zadanie numer 4 polegało na przekształceniu skryptu tak aby samochód zwiększał swoją prędkość z każdą klatką o 0.001 jednostki.

Dodane zostały zmienne przechowujące wartość prędkości i wartość przyśpieszenia.

```
float predkosc;
float przyspieszenie;
```

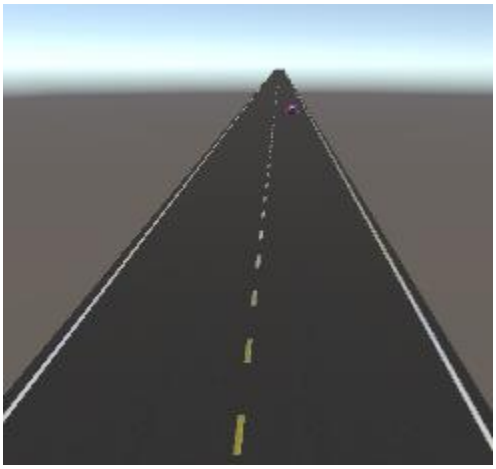
Zainicjalizowanie zmiennych odpowiednimi wartościami.

```
void Start()
    {
        predkosc = 0.0f;
        przyspieszenie = 0.001f;
    }
```

Przerobienie metody Transform aby jako argument Z przyjmowała zmienną predkosc, która będzie co klatkę zwiększana o wartość zmiennej przyspieszenie.

```
void Update()
    {
        transform.Translate(0.0f, 0.0f, predkosc);
        predkosc += przyspieszenie;
    }
```

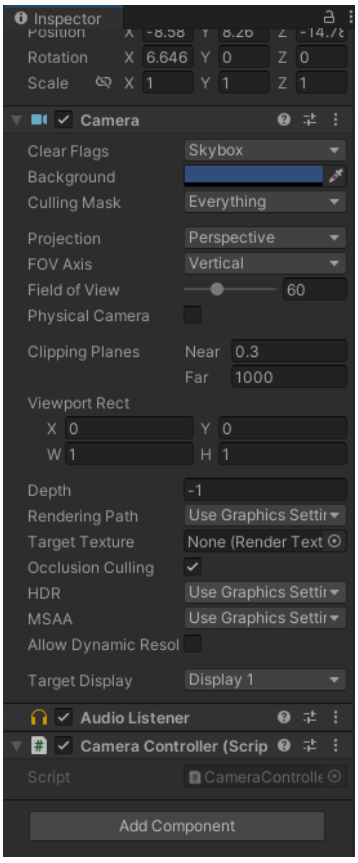
Samochód po uruchomieniu gry cały czas zwiększa swoją prędkość.

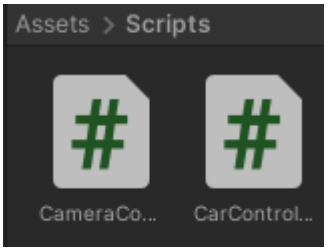


Zadanie 5

Zadanie numer 5 polegało napisaniu skryptu sterowania kamerą. Kamera powinna podążać za samochodem podczas jego ruchu. Widok kamery powinien obejmować tylną część pojazdu i drogę przed nim.

Utworzenie nowego skrytpu o nazwie „CameraController” i dołączenie go do kamery.





Otworzenie skryptu i dodanie do niego zmienną publiczną o nazwie car typu GameObject, która będzie odpowiadać.

```
public GameObject car;
```

Dodanie zmiennej o nazwie cameraPosition typu Vector3, która będzie przechowywać współrzędne położenia kamery.

```
Vector3 cameraPosition;
```

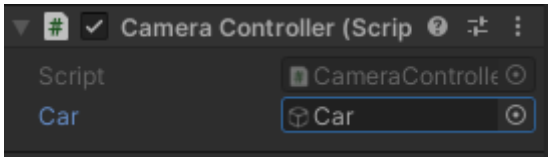
Przypisanie pozycji kamery pozycji samochodu w metodzie Update.

```
cameraPosition = car.transform.position;
```

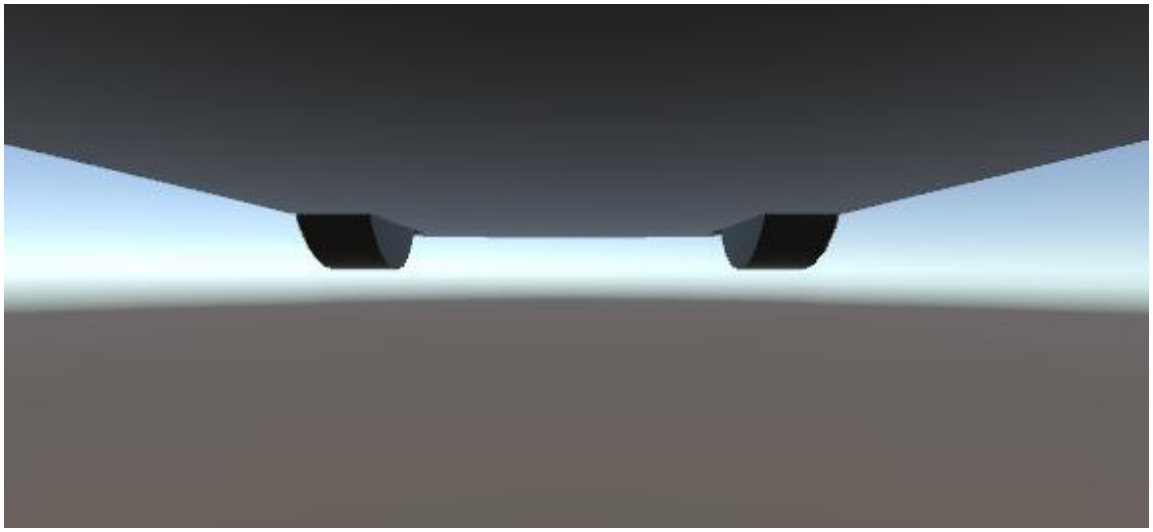
Przypisanie uzyskanej pozycji kamery do jej komponentu.

```
transform.position = cameraPosition;
```

Zapisanie skryptu, otworzenie okna Inspector obiektu kamery. W komponencie skryptu w panelu Car zaznaczony został obiekt samochodu ze sceny.



Po uruchomieniu gry kamera śledzi pojazd, jednak jest w środku pojazdu więc należy lekko przesunąć współrzędne poprzez dodanie przykładowych wartości do odpowiednich współrzędnych. Aby kamera była wyżej należy dodać dodatnią wartość do wartości współrzędnej Y. Ponadto kamera powinna być trochę z tyłu pojazdu, co odpowiada dodaniu ujemnej wartości do wartości współrzędnej Z.



Kod po modyfikacjach.

```
cameraPosition.y += 5.0f;
cameraPosition.z -= 10.0f;
```

Efekt.

Cały kod.

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class CameraController : MonoBehaviour
{
    public GameObject car;
    Vector3 cameraPosition;

    // Start is called before the first frame update
    void Start()
    {

    }

    // Update is called once per frame
    void Update()
    {
        cameraPosition = car.transform.position;
        cameraPosition.y += 5.0f;
        cameraPosition.z -= 10.0f;
        transform.position = cameraPosition;

    }
}
```