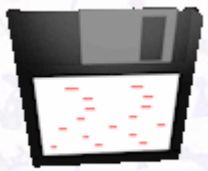
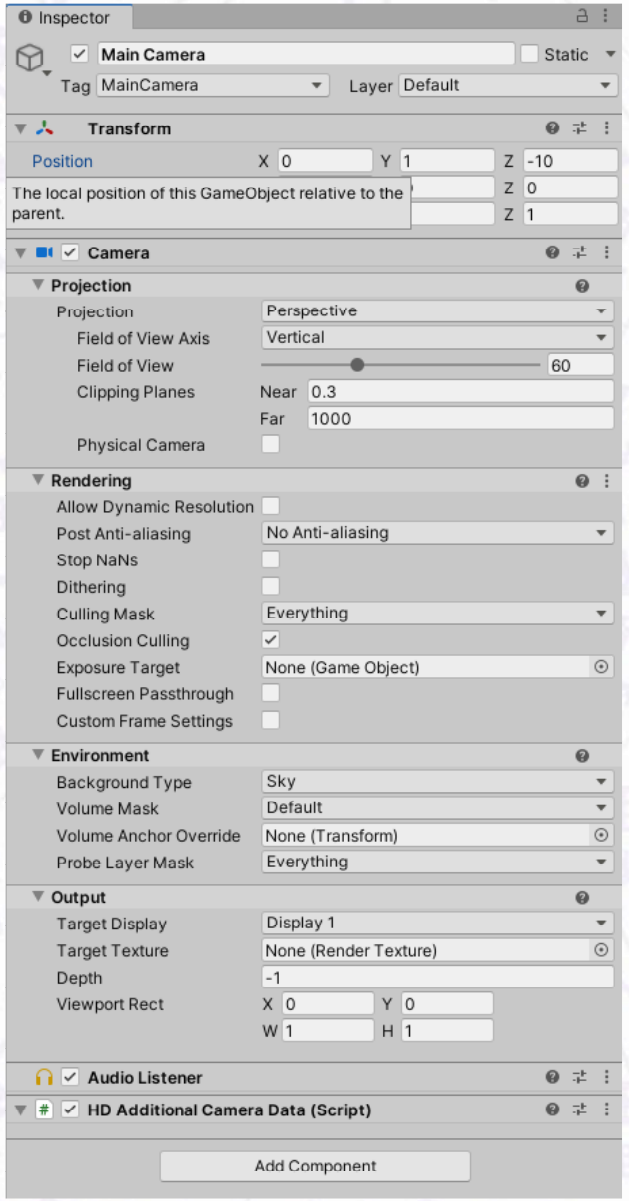
PROGRAMOWANIE GIER

MODEL KOMPONENTOWY I TWORZENIE SKRYPTÓW

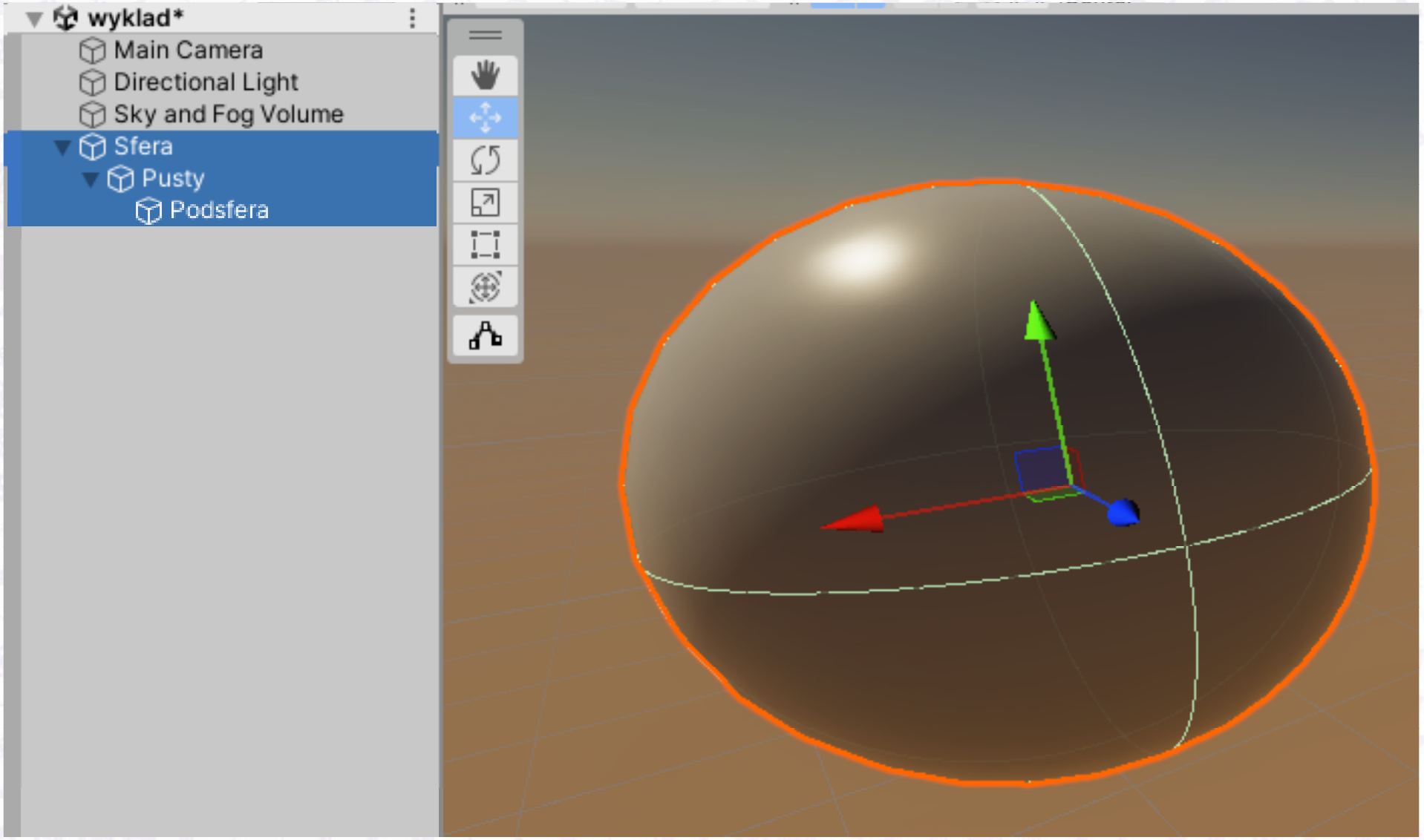
PRZYPOMNIENIE

- Każdy obiekt składa się z komponentów
- Komponent jest klasą, która dziedziczy po klasie GameObject zdefiniowanej w aparacie silnika Unity
- Właściwości komponentów oraz ich wartości możemy zaobserwować w inspektorze
- Każdy komponent ma ściśle określone zadanie
- Klasa GameObject posiada metody, które są wywoływane automatycznie przez system zdarzeń aparatu Unity w momencie zajścia konkretnych zdarzeń
- Komponenty mogą się komunikować z innymi komponentami, które są przypisane do obiektu, a także z komponentami w innych bytach
- Każdy obiekt ma przynajmniej jeden komponent (Transform)





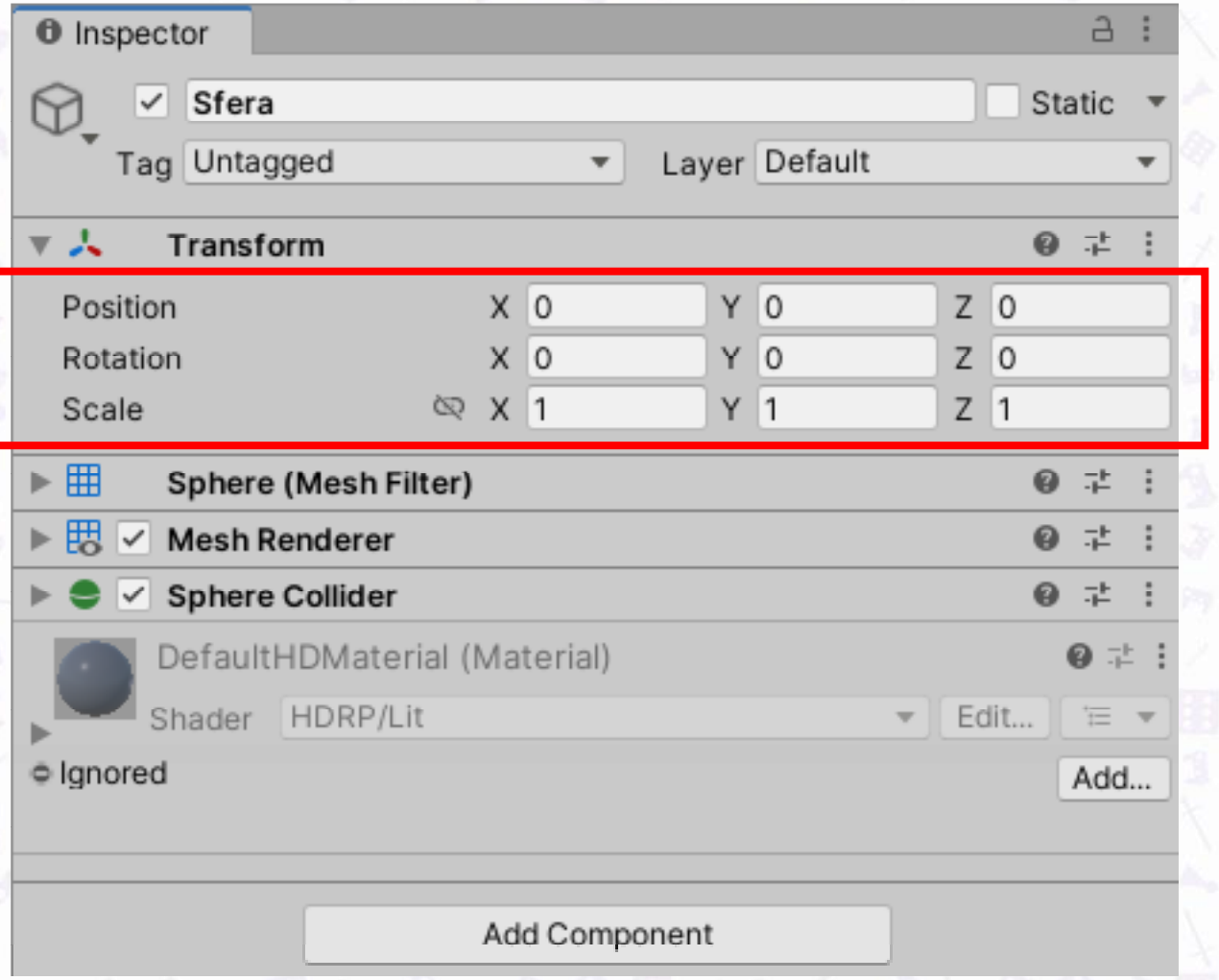
Scena testowa



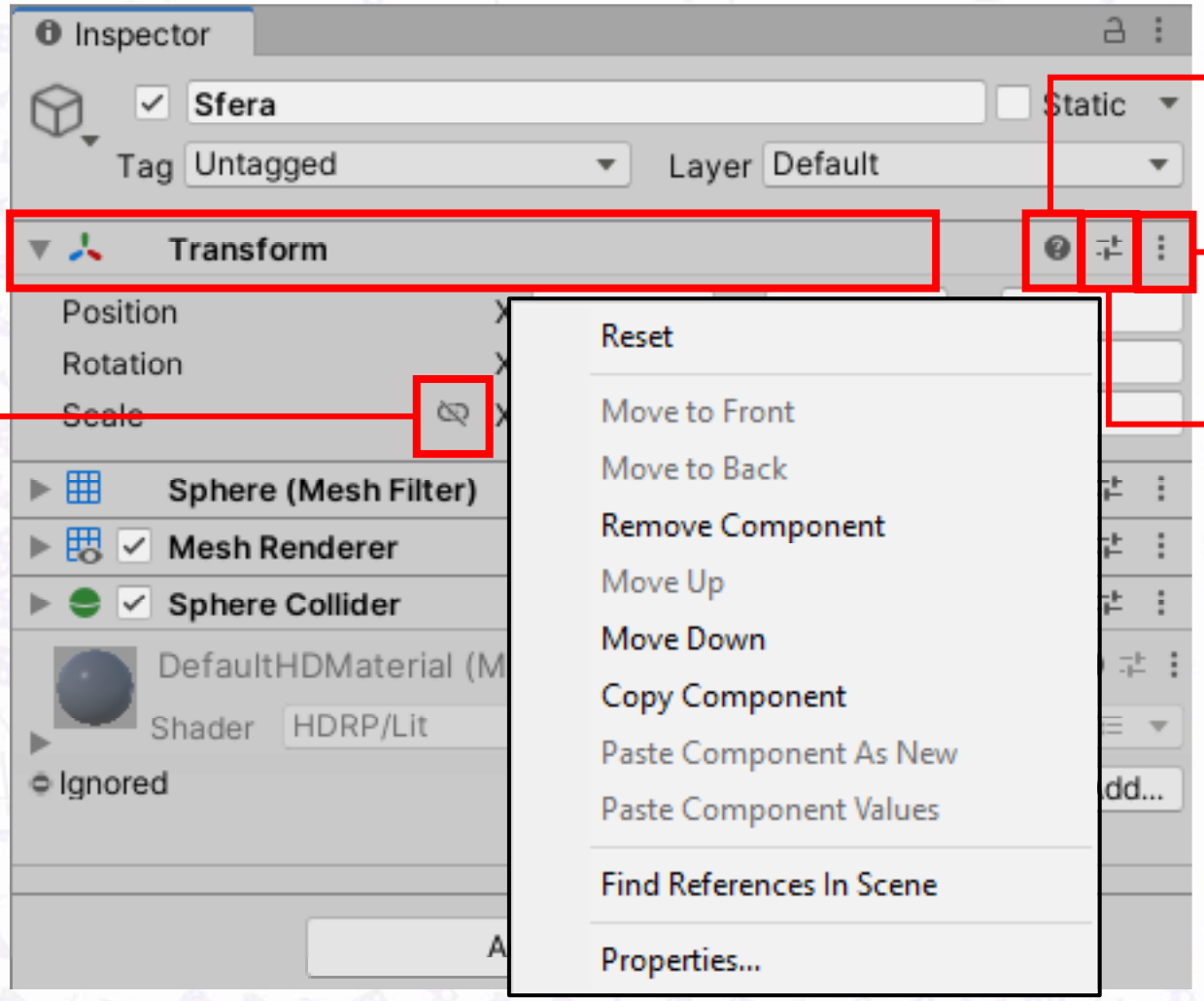
KOMPONENTY SFERY

Byt „Sfera” ma cztery komponenty

1. Transform
2. Sphere (Mesh Filter)
3. Mesh Renderer
4. Sphere Collider



Obsługa komponentów



Zwijanie/rozwijanie

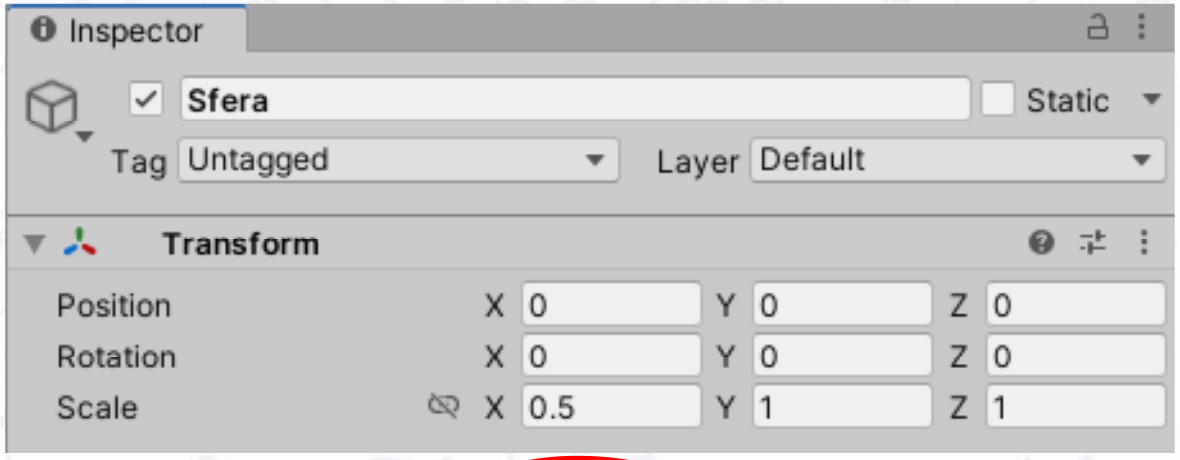
Przełączenie trybu skalowania

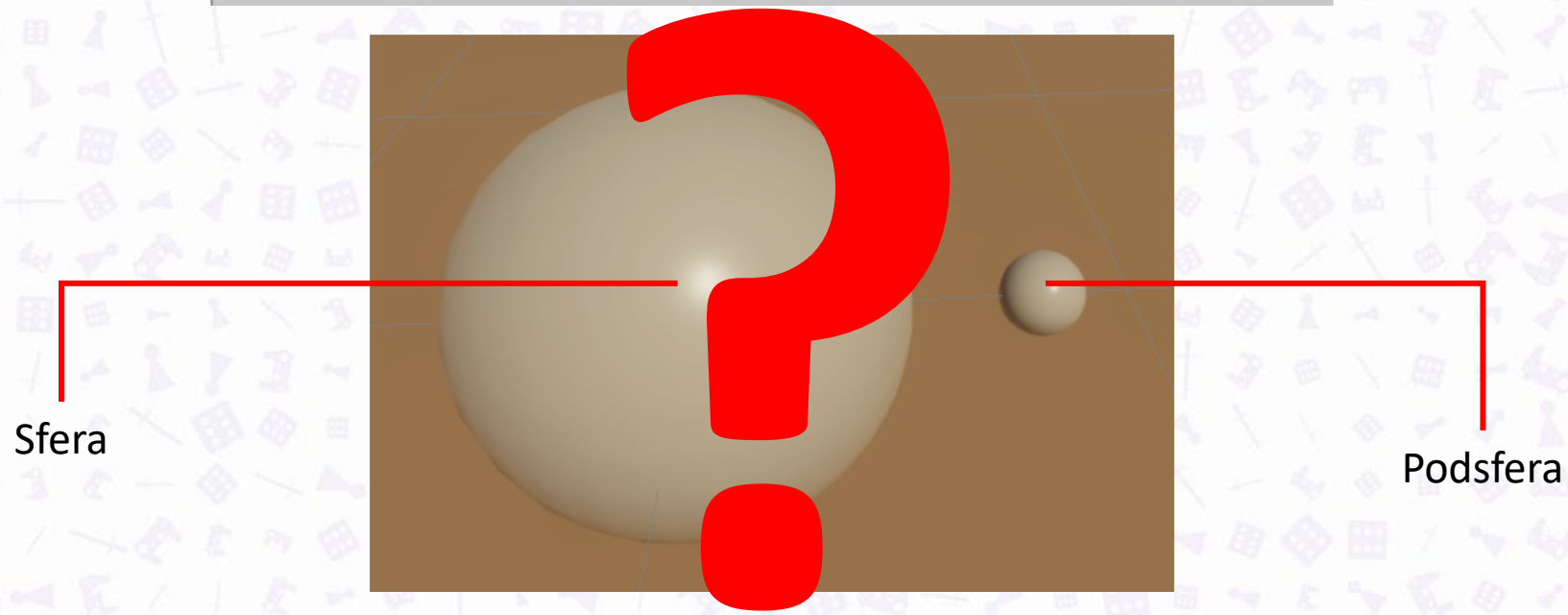
Przejście do dokumentacji

Menu kontekstowe komponentu

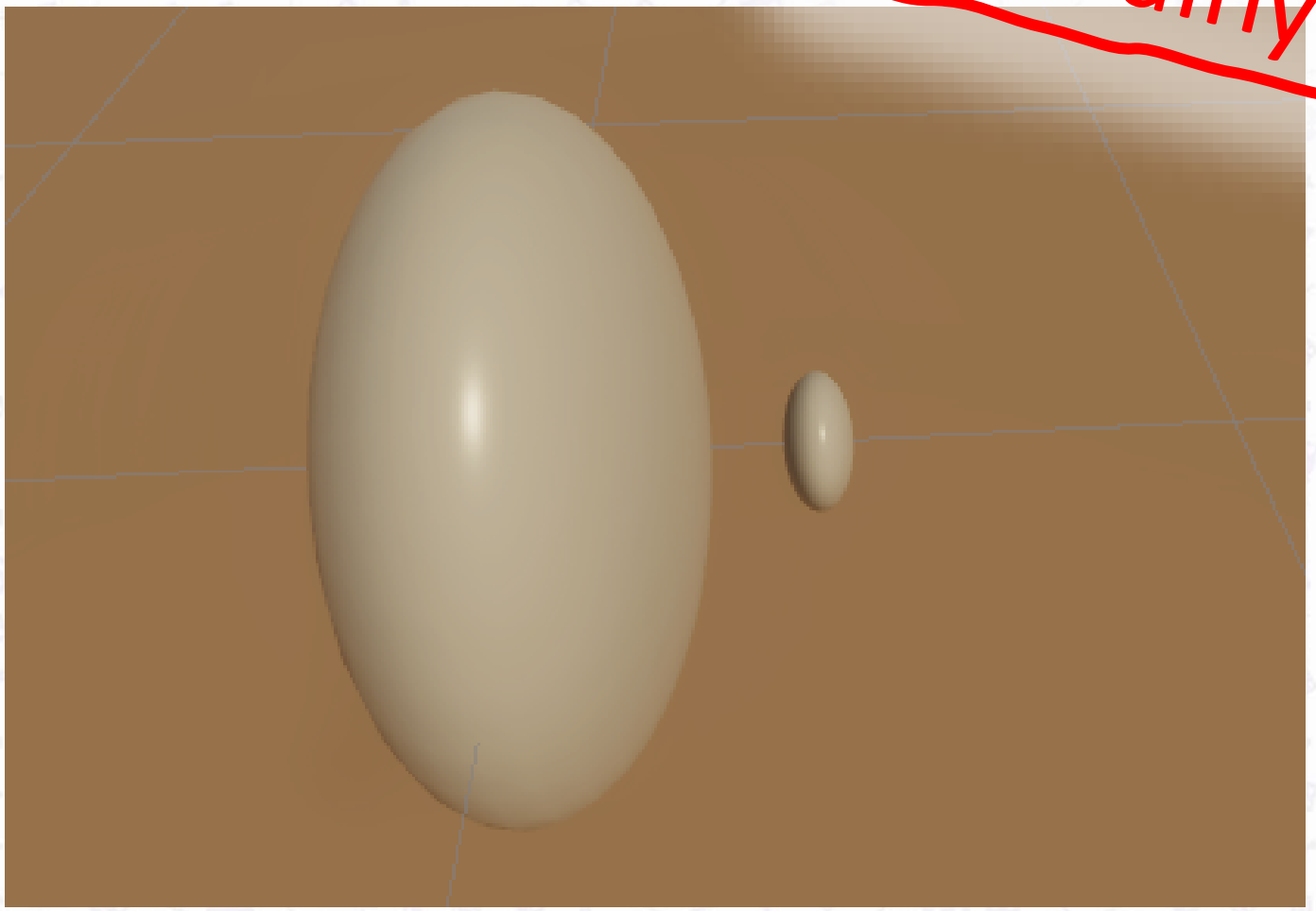
Zapis lub załadowanie wartości

TRANSFORM: UKŁAD WSPÓŁRZĘDNYCH?



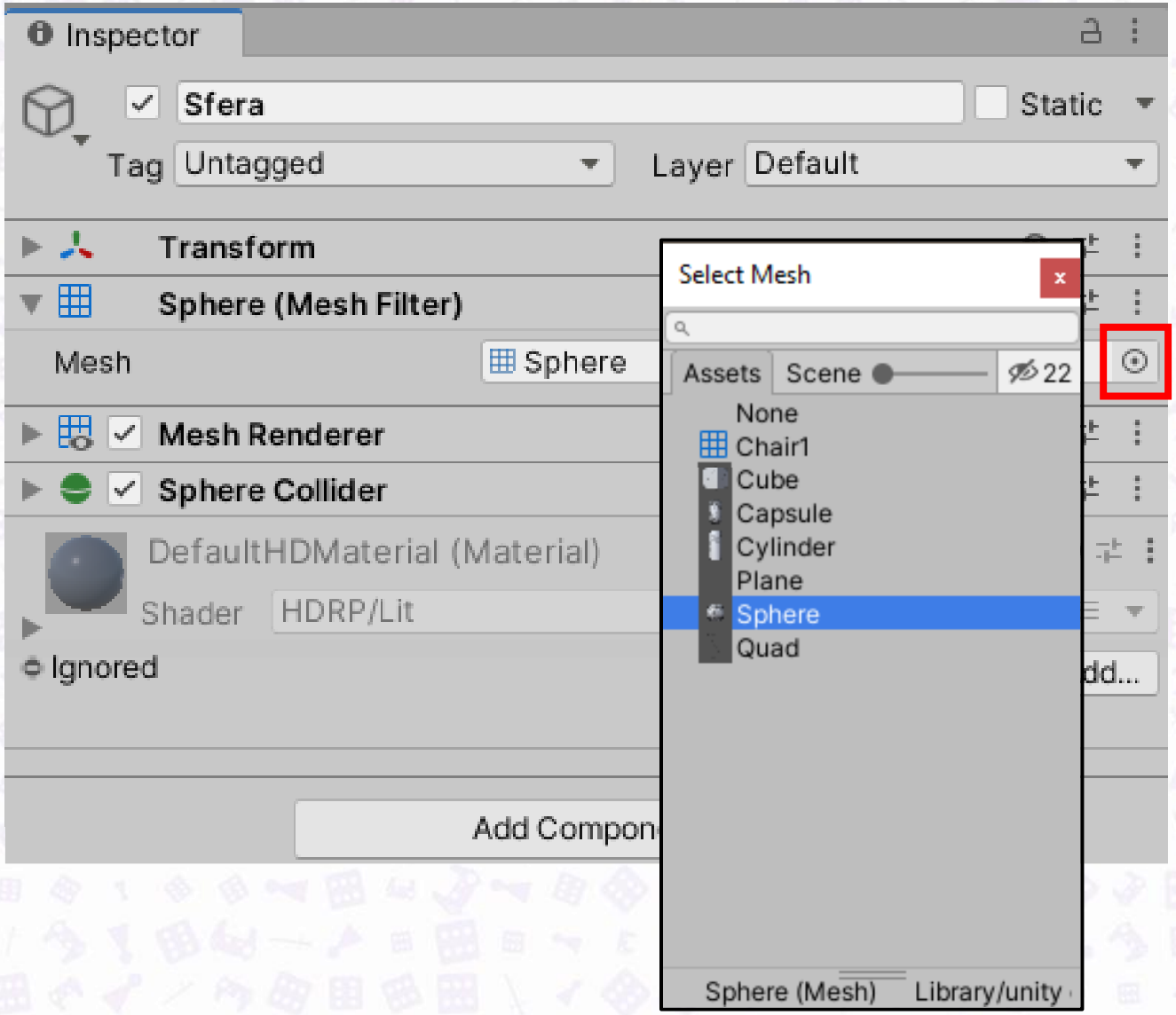


TRANSFORM: UKŁAD WSPÓŁRZĘDNYCH?



MESH FILTER

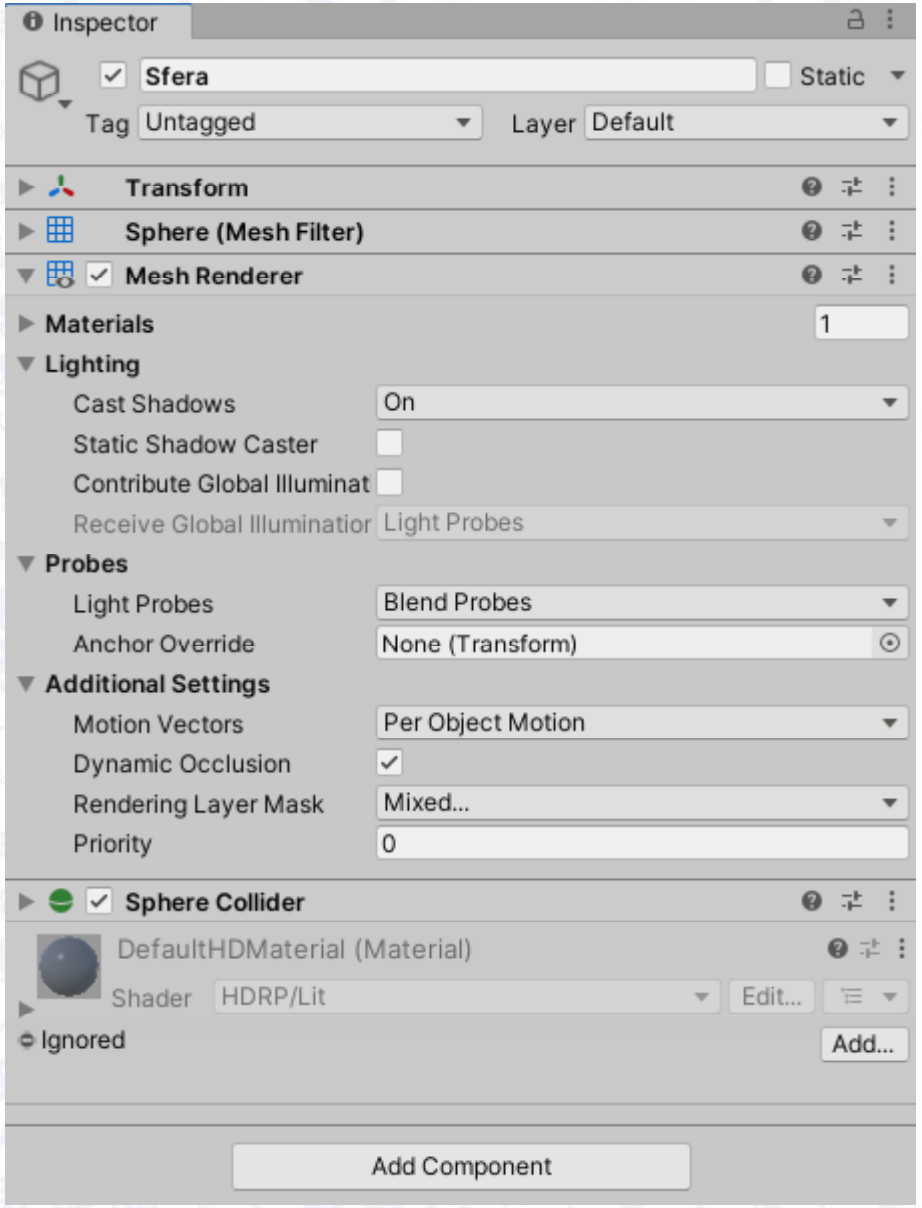
Komponent Mesh Filter przechowuje dane o punktach, krawędziach i wielokątach, z których składa się siatka modelu.



Mesh Renderer

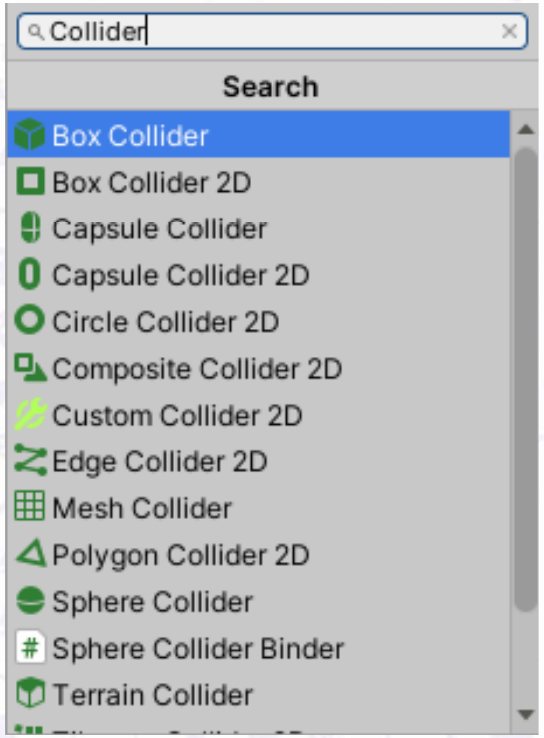
Komponent Mesh Renderer odpowiada za wyświetlanie siatki bytu, do którego jest przypisany. Pozwala również skonfigurować sposób wyświetlania oraz najważniejsze parametry:

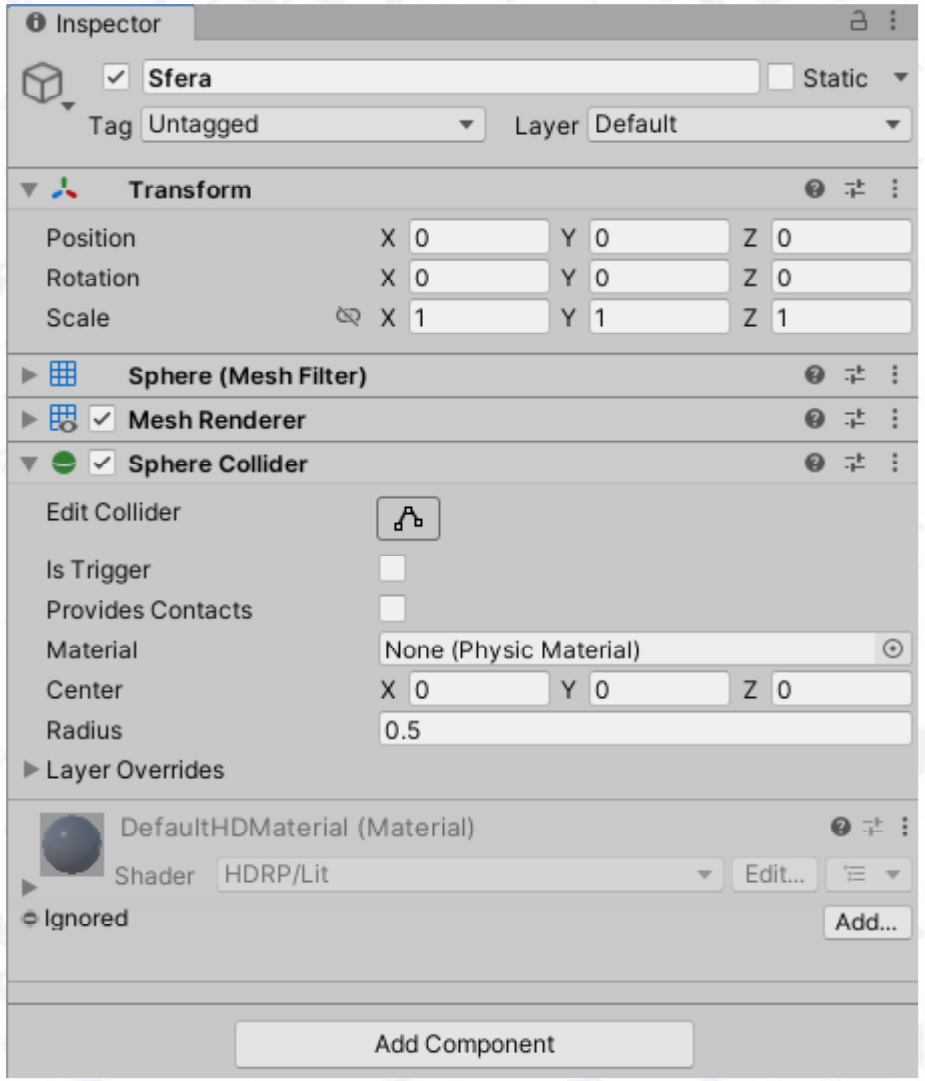
- materiałów
- oświetlenia
- próbników
- pozostałych właściwości



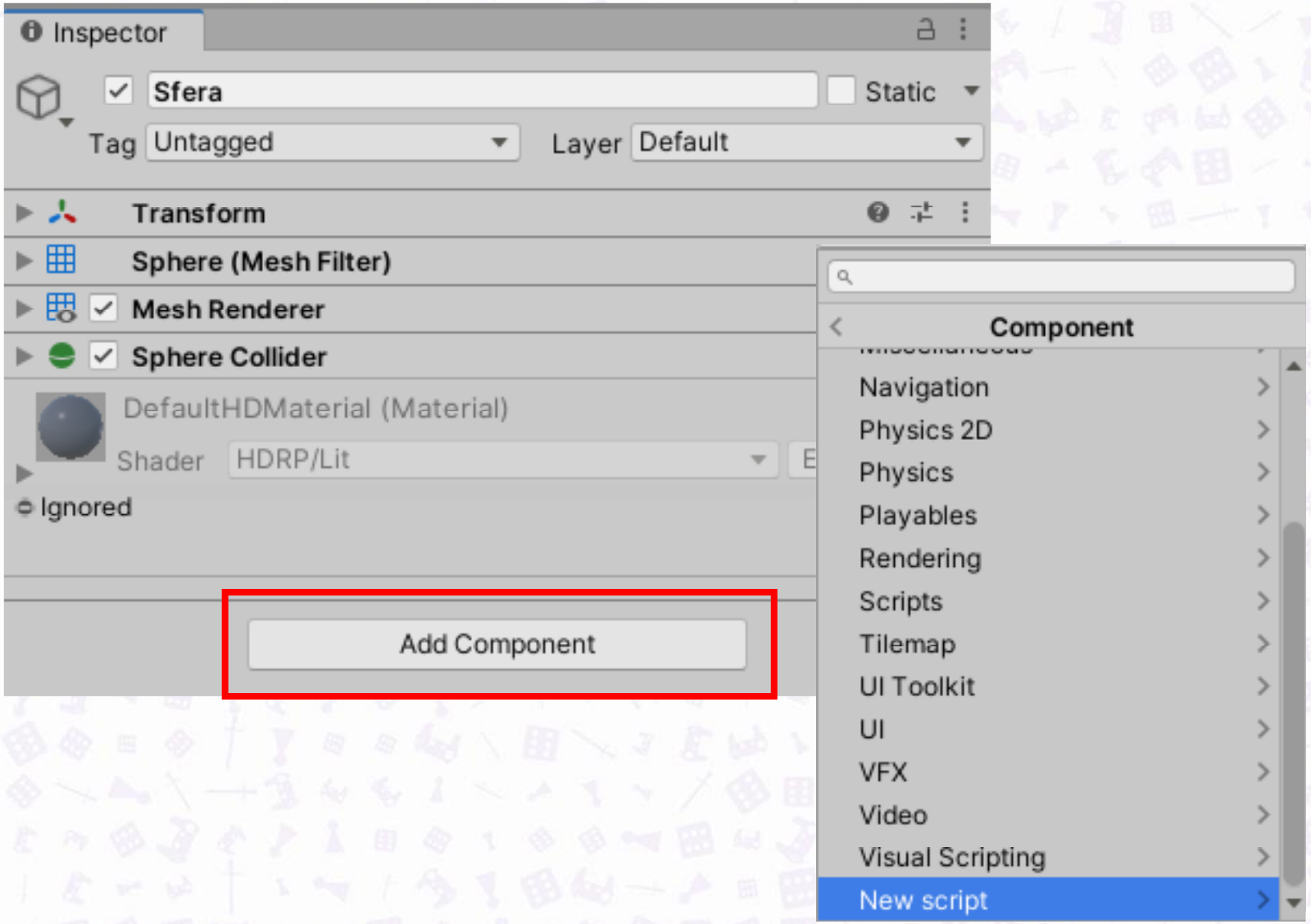
SPHERE COLLIDER

Komponent Sphere Collider definiuje bryłę brzegową bytu („hitbox”). Jest jednym z wielu komponentów, które mają taki cel i które są zoptymalizowane pod konkretny kształt.

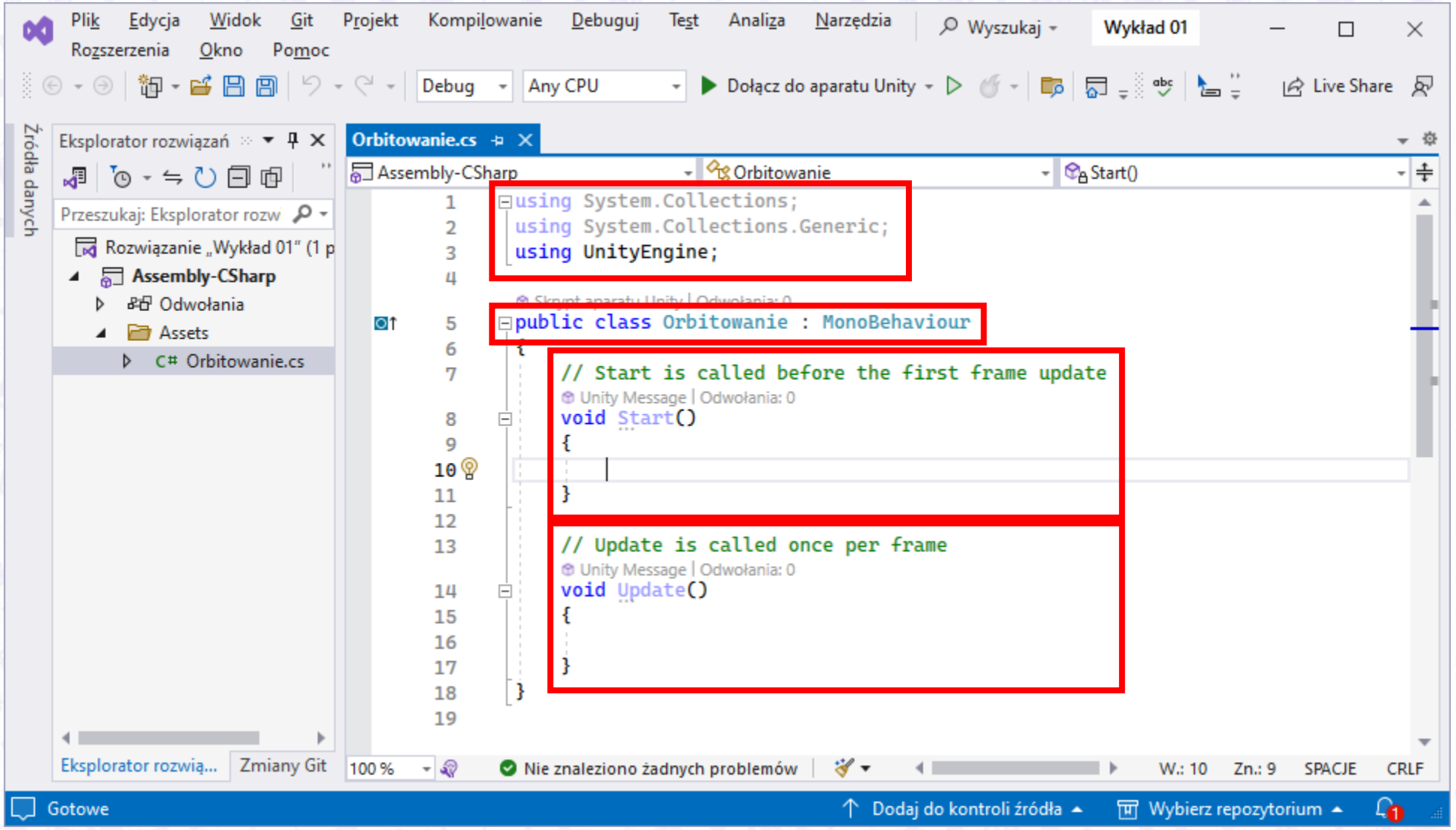




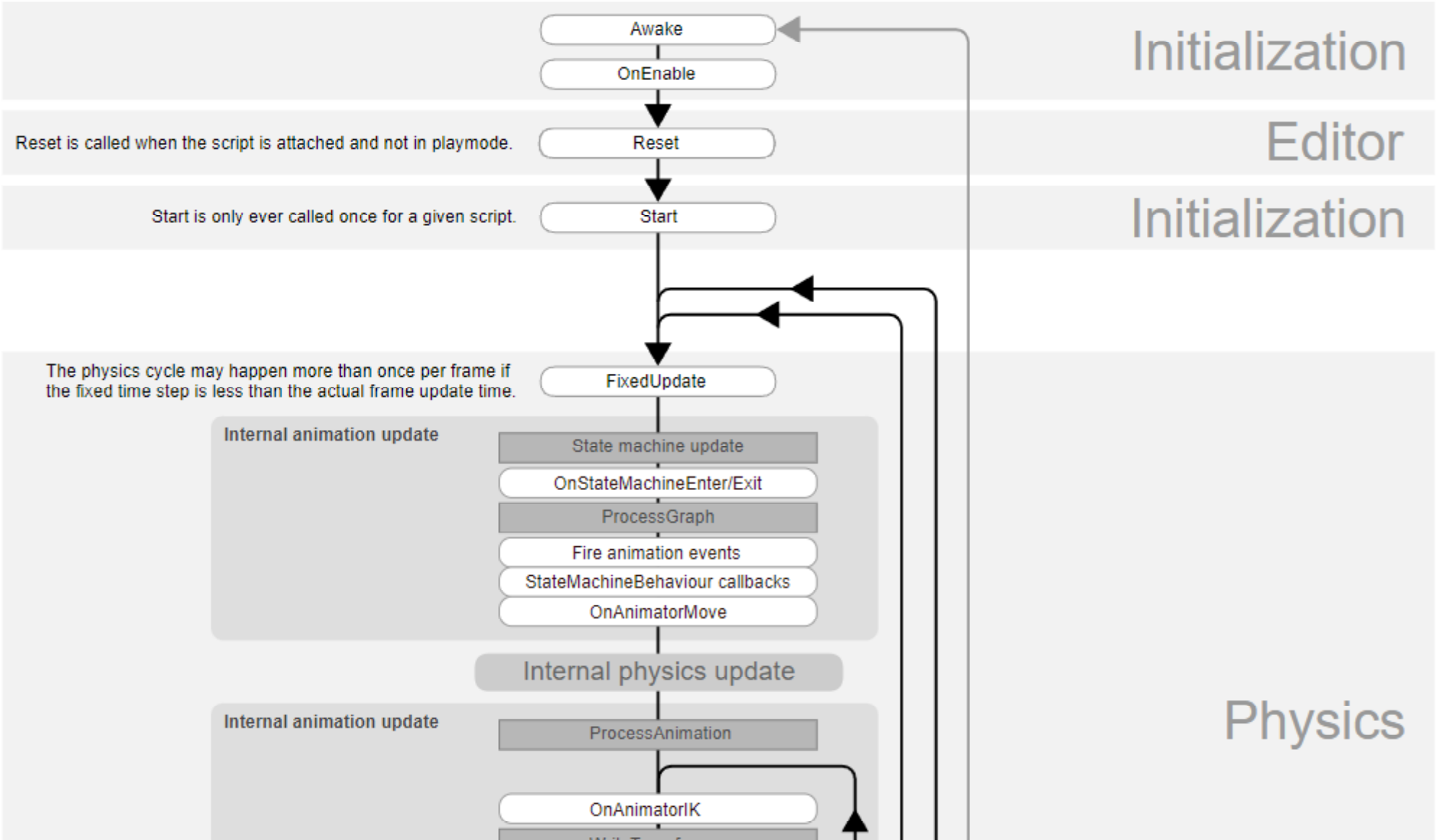
Dodawanie komponentów



PIERWSZY SKRYPT



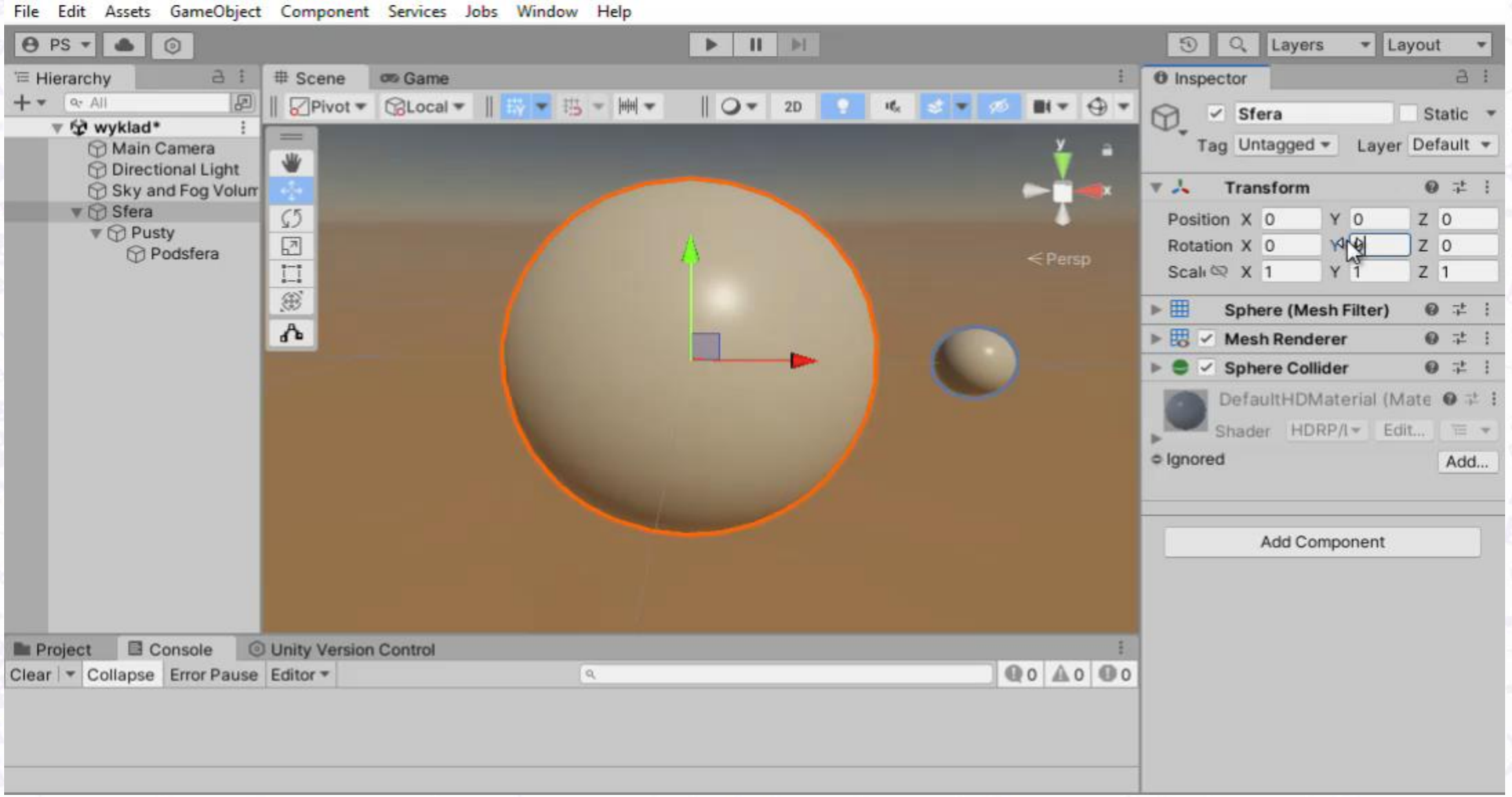
Funkcje aparatu Unity



Funkcje aparatu Unity

Funkcja	Moment wywołania	Do czego wykorzystywana?
Awake	**Jednorazowo**, w momencie aktywacji bytu	Inicjalizacja, pobieranie referencji
Start	**Jednorazowo**, w momencie aktywacji komponentu	Inicjalizacja, nadawanie wartości początkowych
Update	Każdorazowo, podczas przetwarzania klatki animacji	Logika gry, interpretacja sterowania, modyfikacja sceny
FixedUpdate	Każdorazowo, w każdej klatce obliczeń fizycznych	Dodatkowe obliczenia fizyczne, wczytywanie sterowania
OnEnable	Każdorazowo, gdy komponent staje się aktywny	Tworzenie efektów cząsteczkowych, komunikacja z systemami zdarzeń
OnDisable	Każdorazowo, gdy komponent staje się nieaktywny	Tworzenie efektów cząsteczkowych, komunikacja z systemami zdarzeń

ORBITOWANIE: USE CASE



ORBITOWANIE: „ALGORYTM”

1. Pobierz referencję do komponentu Transform
2. W każdej klatce animacji, powtarzaj:
3. Pobierz aktualną wartość rotacji
4. Zwiększ rotację o pewną wartość
5. Wróć do kroku 2

?

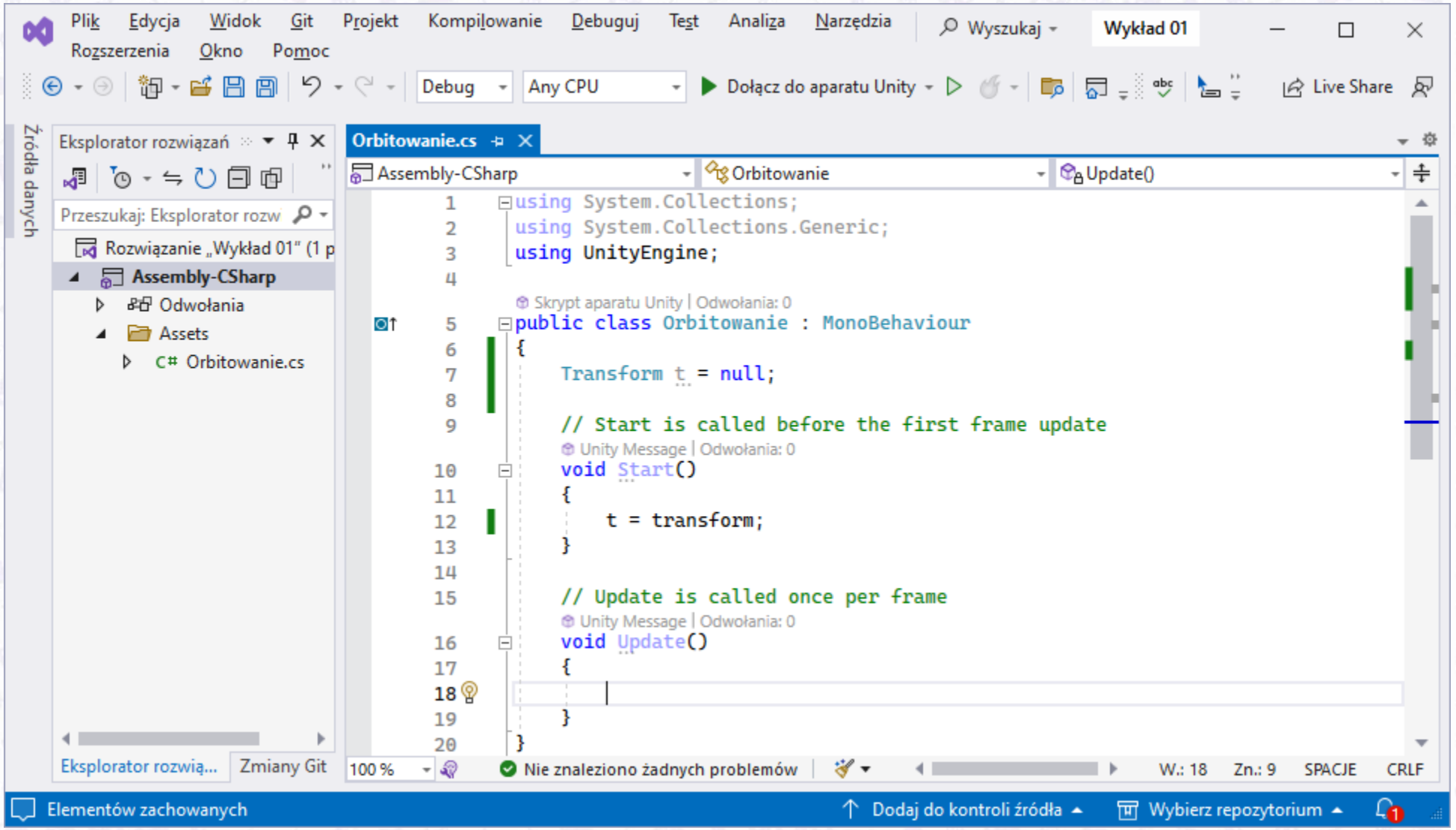
Update!

Odczytać pole klasy…?

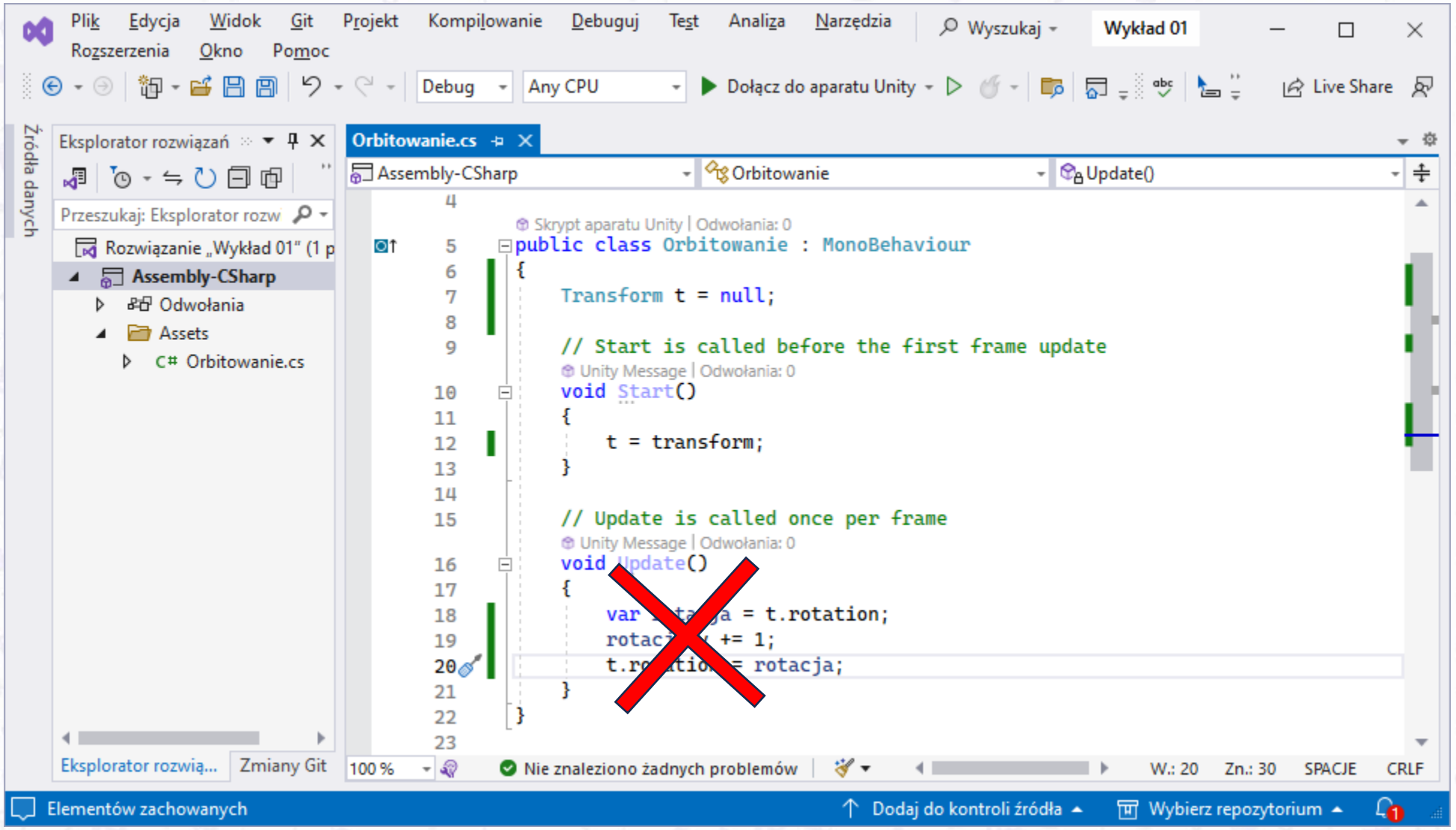
Po prostu dodać…?

Update!

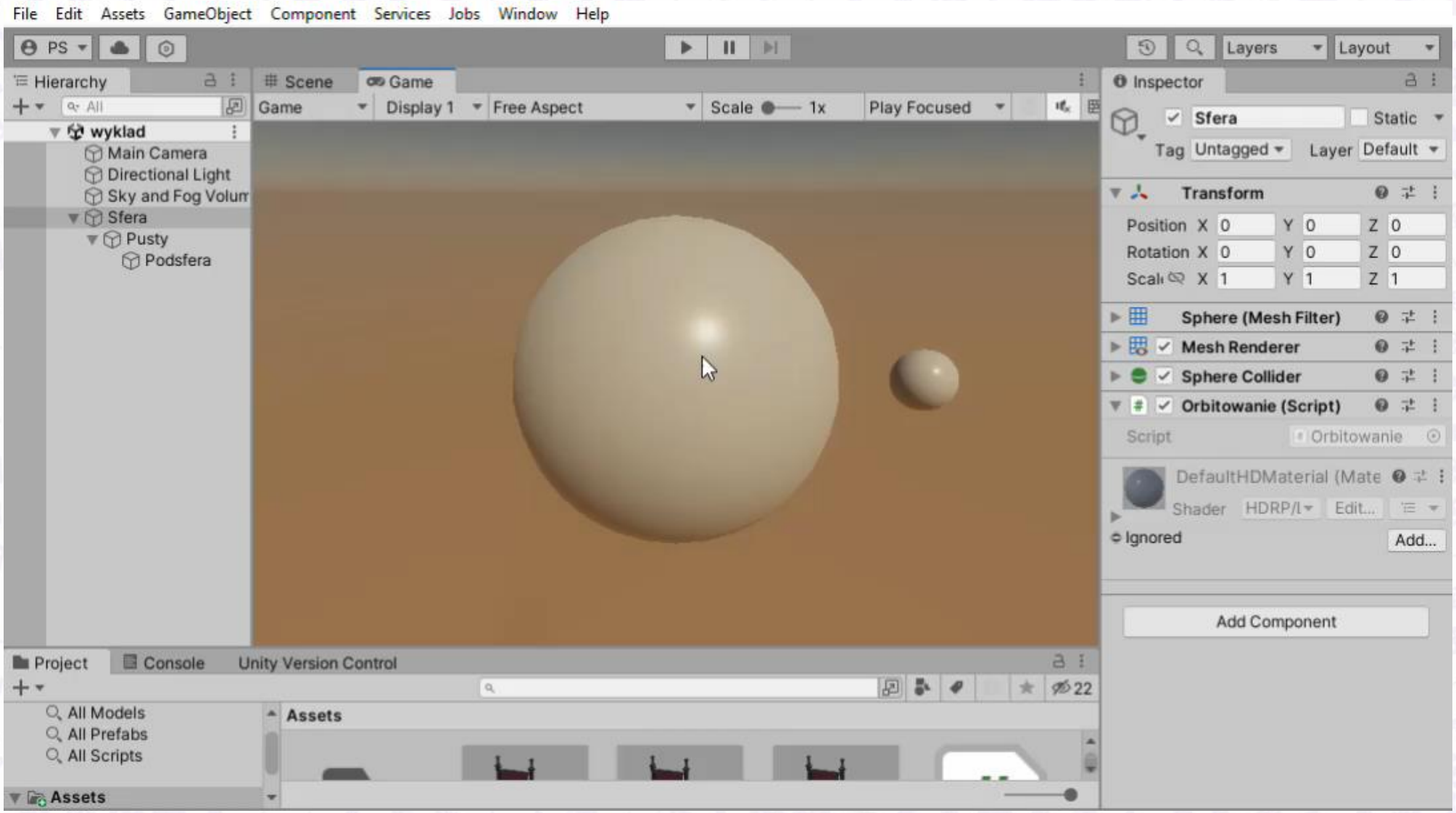
ORBITOWANIE: KROK 1



ORBITOWANIE: KROK 3 I 4



ORBITOWANIE: EFEKT



Co mówi dokumentacja?

Transform.rotation

Leave feedback

SWITCH TO MANUAL

public Quaternion **rotation**;

at stores the rotation of the Transform in world space.

ion stores a Quaternion. You can use rotation to rotate a GameObject or provide
ion. Do not attempt to edit/modify rotation. Transform.rotation is less than 180

ion has no gimbal lock.

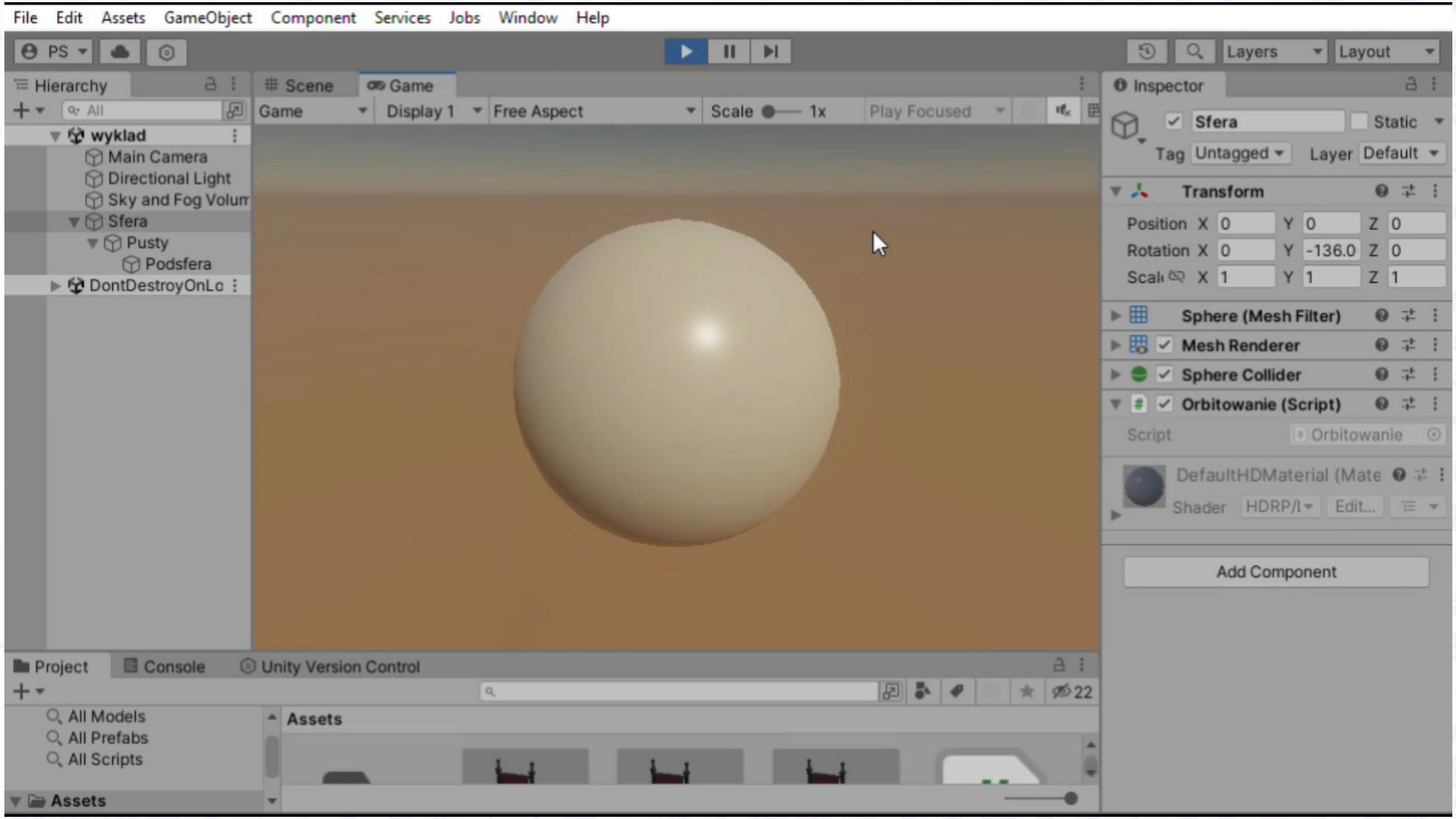
To rotate a Transform, use Transform.Rotate, which uses Euler Angles.

If you want to match values you see in the Inspector, use the Quaternion.eulerAngles property on the returned Quaternion.

ORBITOWANIE: KROK 3 I 4 POPRAWIONE

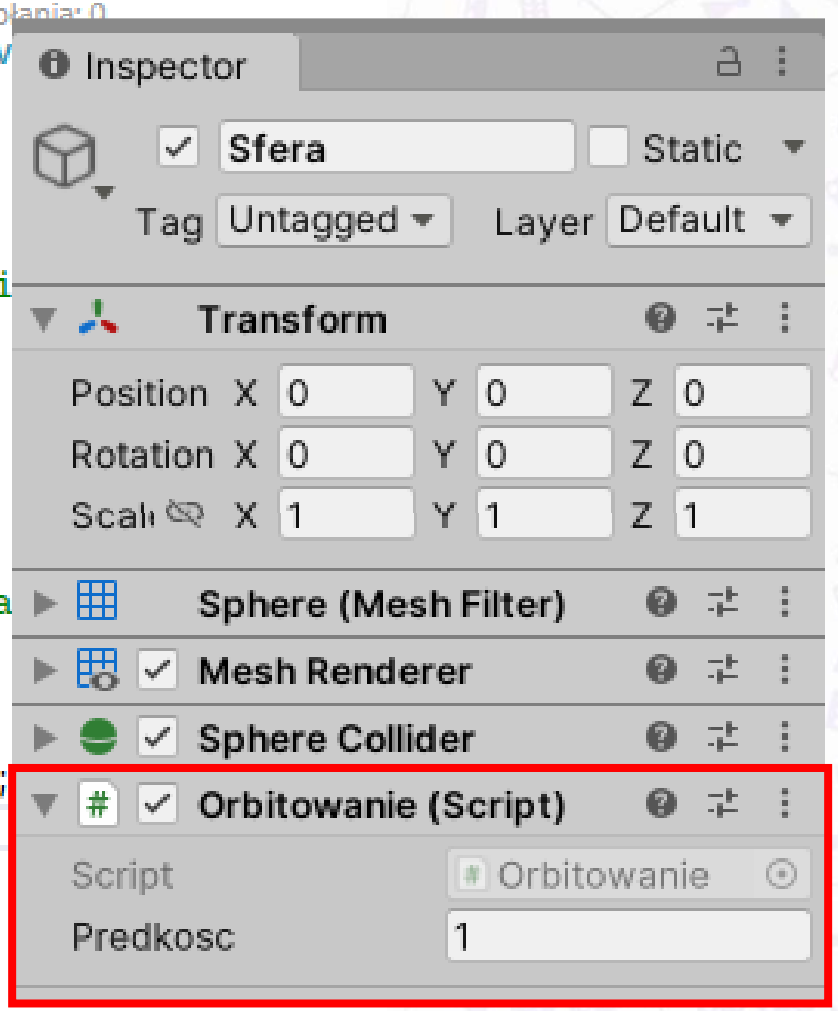
```
Skrypt aparatu Unity (1 odwołanie do zasobu) | Odwołania: 0
5   public class Orbitowanie : MonoBehaviour
6   {
7       Transform t = null;
8
9       // Start is called before the first frame update
            Unity Message | Odwołania: 0
10      void Start()
11      {
12          t = transform;
13      }
14
15      // Update is called once per frame
            Unity Message | Odwołania: 0
16      void Update()
17      {
18          var rotacja = t.eulerAngles;
19          rotacja.y += 1;
20          t.eulerAngles = rotacja;
21      }
22  }
23
```

ORBITOWANIE: EFEKT

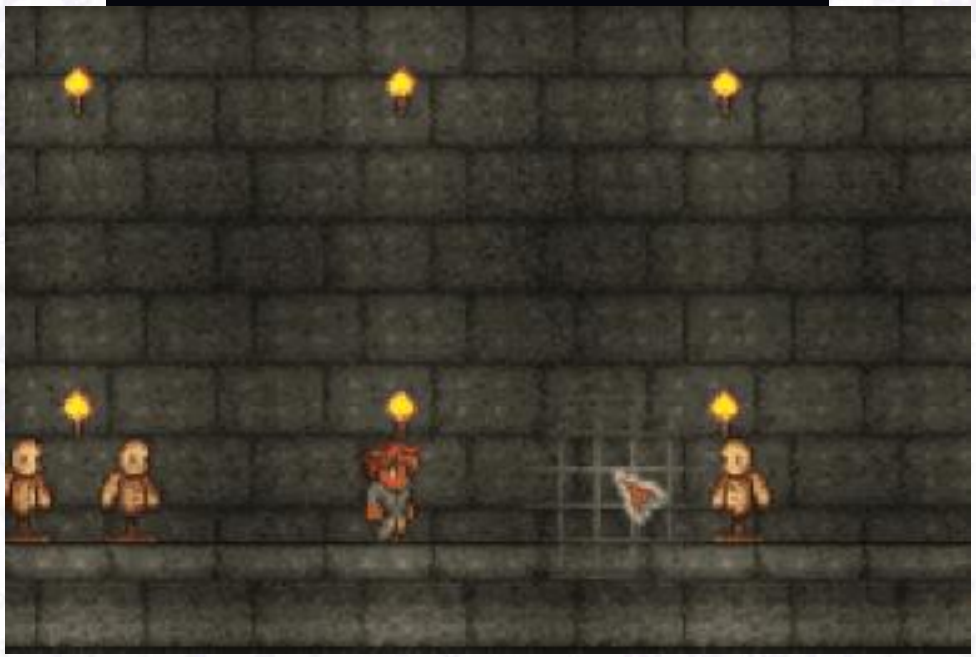


ORBITOWANIE: ZMIENNA PREDKOSC

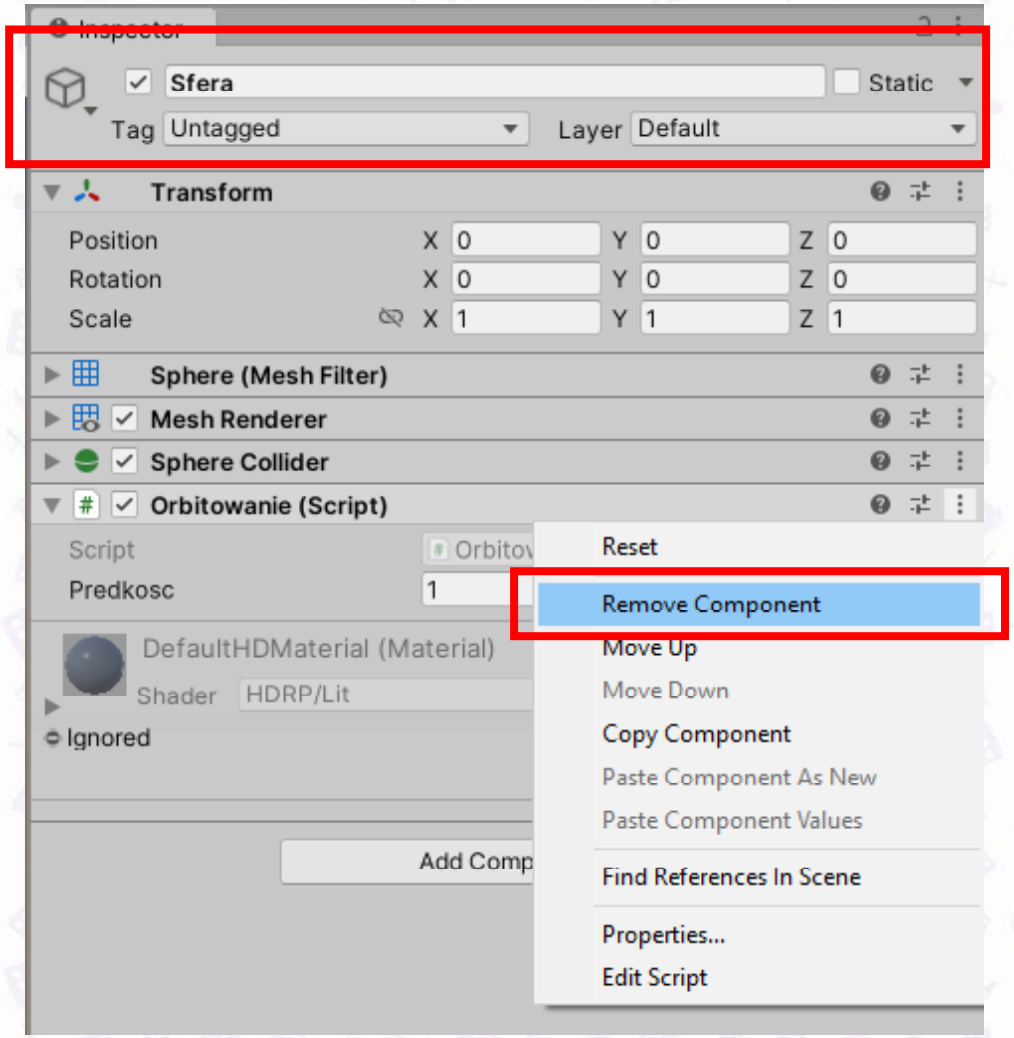
```
Skrypt aparatu Unity (1 odwołanie do zasobu) | Odwołania: 0
5   public class Orbitowanie : MonoBehav
6   {
7       Transform t = null;
8       public float predkosc = 1f;
9
10      // Start is called before the fi
            Unity Message | Odwołania: 0
11      void Start()
12      {
13          t = transform;
14      }
15
16      // Update is called once per fra
            Unity Message | Odwołania: 0
17      void Update()
18      {
19          var rotacja = t.eulerAngles;
20          rotacja.y += predkosc;
21          t.eulerAngles = rotacja;
22      }
23  }
24
```

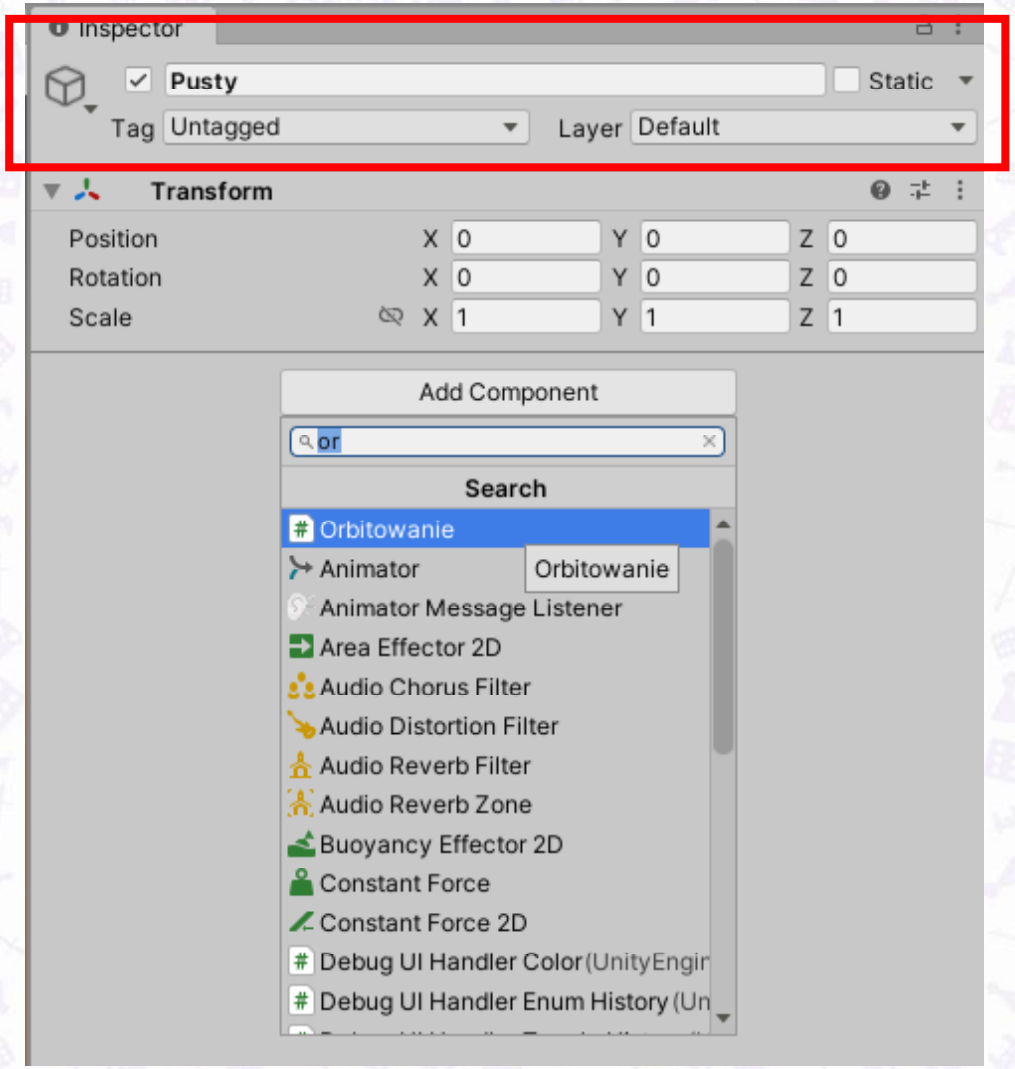


ORBITOWANIE: PRZYDATNE?

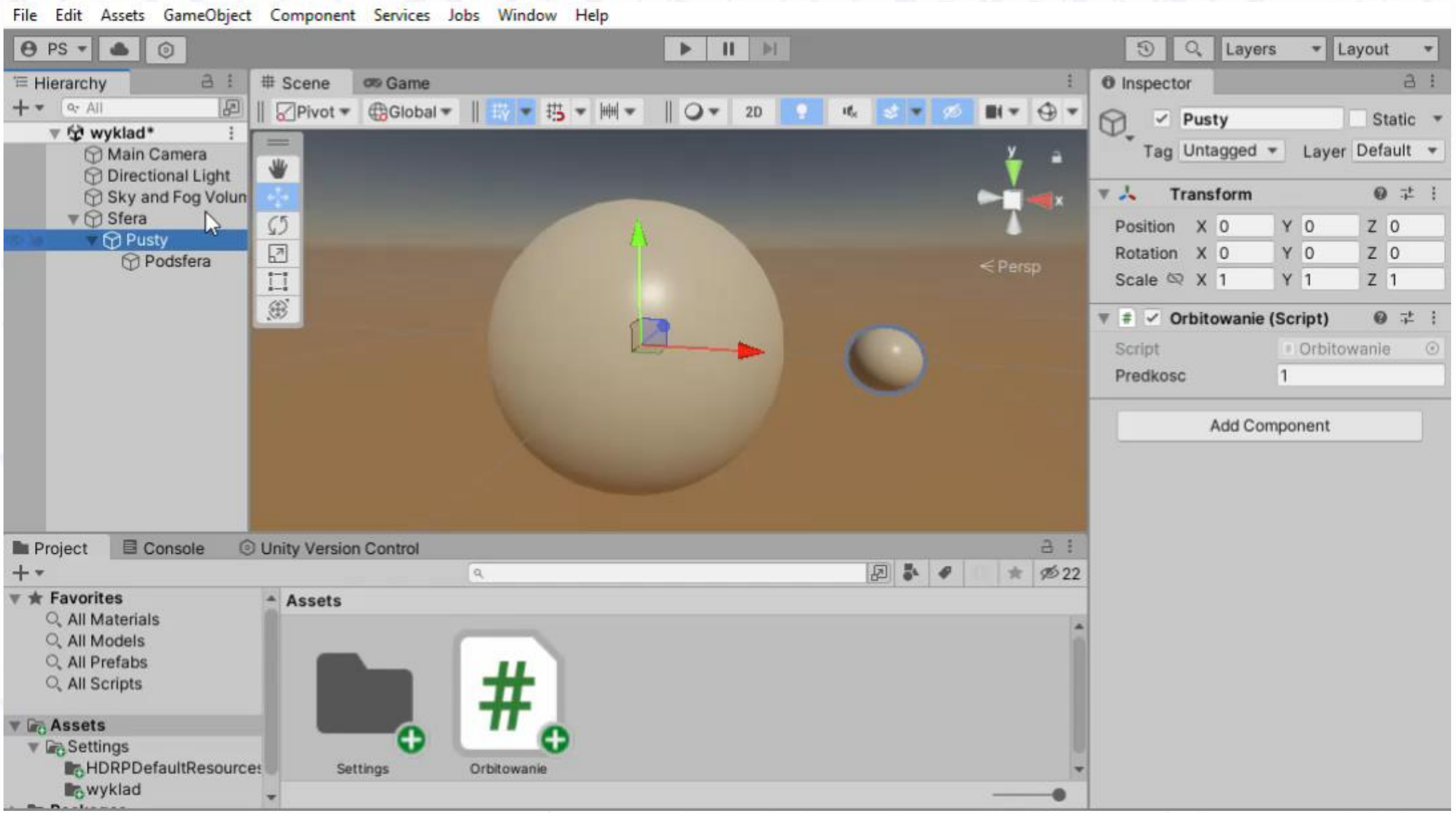


PRZENIESIENIE SKRYPTU





UTWORZENIE PREFABRYKATU



INSTANTIATE

Składnia: public static Object Instantiate(Object oryginał, Transform rodzic)

Działanie: Tworzy kopię obiektu podanego jako pierwszy argument, ustawia jego rodzica na obiekt podany jako drugi argument, a następnie zwraca referencję do utworzonego obiektu.

Dokumentacja: https://docs.unity3d.com/ScriptReference/Object.Instantiate.html

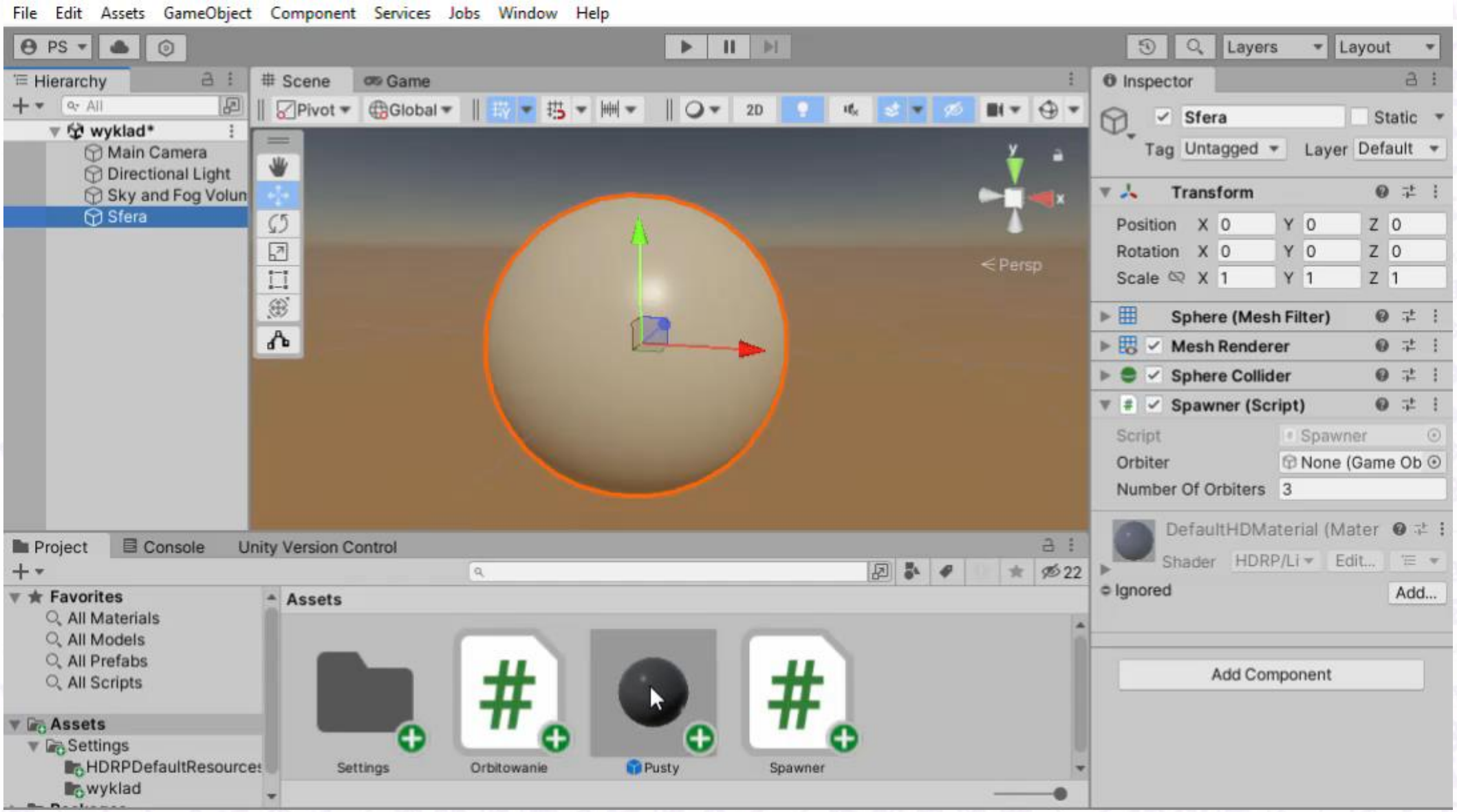
SKRYPT TWORZĄCY PODSFERY

```
Skrypt aparatu Unity | Odwołania: 0
public class Spawner : MonoBehaviour
{
    public GameObject orbiter;
    public int numberOfOrbiters = 3;

    // Start is called before the first frame update
    Unity Message | Odwołania: 0
    void Start()
    {
        for (int i = 0; i < numberOfOrbiters; i++)
        {
            var nowyObiekt = Instantiate(orbiter, this.transform);

            var rotacja = nowyObiekt.transform.eulerAngles;
            rotacja.y = 360f / numberOfOrbiters * i;
            nowyObiekt.transform.eulerAngles = rotacja;
        }
    }
}
```

SKRYPT TWORZĄCY PODSFERY



Dziękuję za uwagę

Proszę o pytania