

A misty forest scene with tall, slender trees and a path. Numerous glowing yellow light spheres of various sizes are scattered throughout the scene, creating a magical atmosphere. The text is overlaid on a black rectangular background on the right side.

engine UNITY

KAMIL CECHERZ

ŁUKASZ BEDNARSKI

Czym jest Unity ?

Unity – zintegrowane środowisko do tworzenia trójwymiarowych oraz dwuwymiarowych gier komputerowych lub innych materiałów interaktywnych, takich jak wizualizacje czy animacje. Działa na systemie operacyjnym Microsoft Windows, macOS oraz Linux i pozwala tworzyć aplikacje na:

- przeglądarki internetowe,
- komputery osobiste,
- konsole gier wideo
- urządzenia mobilne.



Deweloperzy	Unity Technologies
Pierwsze wydanie	8 czerwca 2005 r . ; 14 lat temu
Wersja stabilna	2019.2.13 / 20 listopada
Wersja zapoznawcza	2019.3.0b12 / 21 listopada 2019 ; 6 dni temu
Napisane w	•C ++ (środowisko wykonawcze) •C # (Unity Scripting API)
System operacyjny	Microsoft Windows , macOS , Linux (wersja zapoznawcza)
Platforma	
Dostępne w	język angielski
Rodzaj	Silnik gry
Licencja	Oprogramowanie prawnie zastrzeżone



Technologie IT a Unity

Językami dopuszczalnymi w Unity są UnityScript (podobieństwo do JavaScript), C# oraz Boo.

Kompatybilne z Unity są hełmy rzeczywistości wirtualnej takie jak Oculus Rift czy Samsung Gear VR.

Gry po zainstalowaniu wtyczki Unity Web Player mogą działać w przeglądarce internetowej.

Obsługiwane platformy

Platformy obejmują iOS, Android, Tizen, Windows, Universal Windows Platform, Mac, Linux, WebGL, PlayStation 4, PlayStation Vita, Xbox One, 3DS, Oculus Rift, Google Cardboard, Steam VR, PlayStation VR, Gear VR, Windows Mixed Reality, Daydream, Android TV, Samsung Smart TV, tvOS, Nintendo Switch, Fire OS, Facebook Gameroom, Apple ARKit, Google ARCore, Vuforia i Magic Leap.

AI vs Unity

Unity Machine Learning Agents to oprogramowanie typu open source, dzięki któremu platforma Unity łączy się z programami uczenia maszynowego, w tym **TensorFlow** firmy Google. Wykorzystując metodę prób i błędów w **Unity Machine Learning Agents**, wirtualne postacie wykorzystują naukę wspomagającą do budowania kreatywnych strategii w realistycznych wirtualnych krajobrazach. Oprogramowanie służy do opracowywania robotów i samochodów samobieżnych.

Model biznesowy Unity

Model licencjonowania

Przez pierwsze dziesięć lat istnienia produktu płatne wersje Unity były sprzedawane wprost; w **2016** roku korporacja zmieniła się na model subskrypcji. Unity ma bezpłatne i płatne opcje licencjonowania. Bezpłatna licencja jest przeznaczona do użytku osobistego lub mniejszych firm generujących mniej niż **100000 USD rocznie**, a subskrypcje są oparte na przychodach generowanych przez gry korzystające z Unity.

Unity Asset Store

Twórcy mogą opracowywać i sprzedawać zasoby generowane przez użytkowników innym twórcom gier za pośrednictwem **Unity Asset Store**. Obejmuje to zasoby i środowiska 3D i 2D dla deweloperów do kupowania i sprzedawania. W **2018** roku sklep dodał bibliotekę przechwytywania ruchu Rokoko. **Unity Asset Store** uruchomiono w **2010** r. Do **2018** r. Za pośrednictwem sklepu cyfrowego pobrano około **40 milionów pobrań**.

Interfejs użytkownika

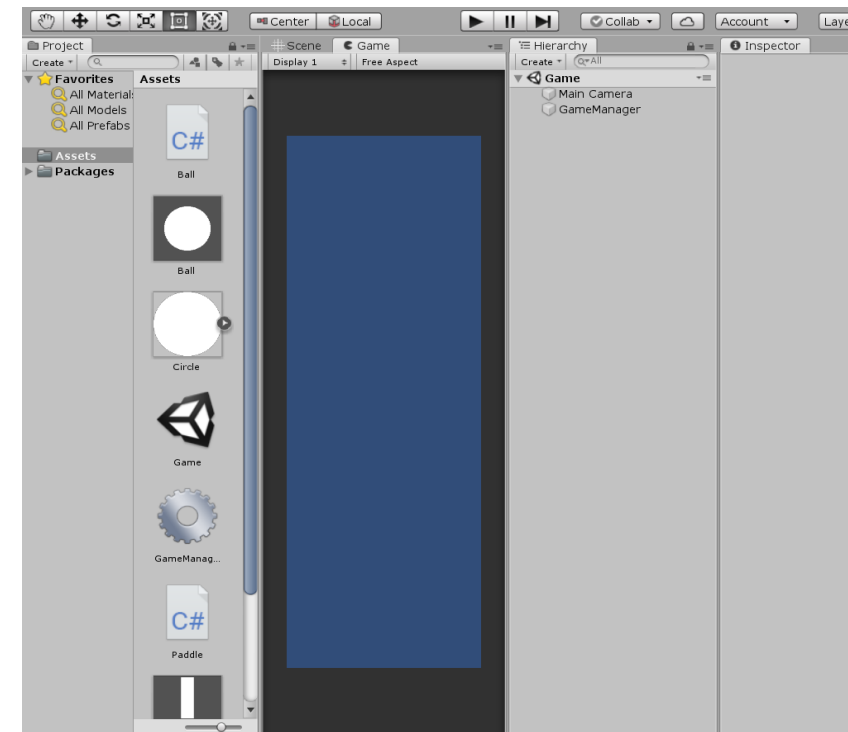
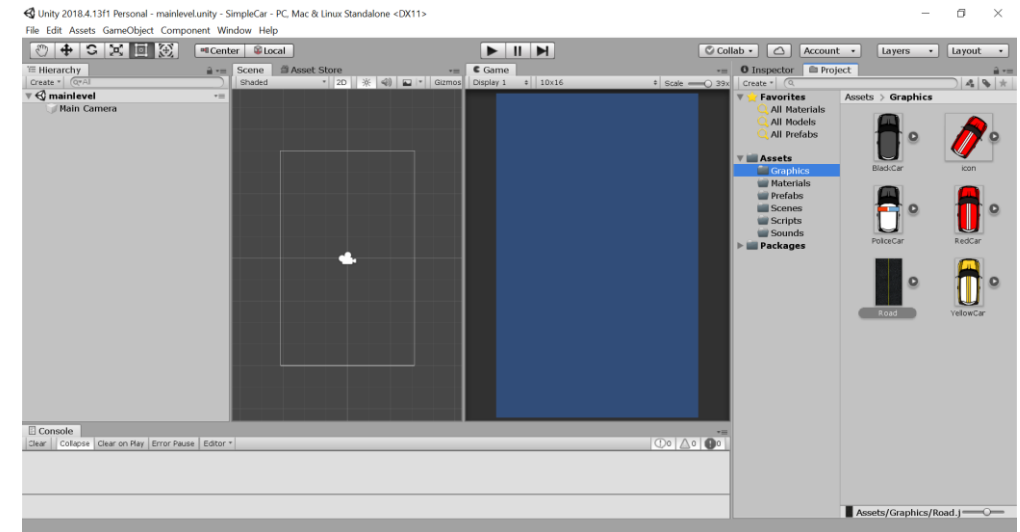
Inspector (inspektor elementu) – Umożliwia dostęp do wszystkich właściwości każdego elementu w grze. Można je dowolnie identyfikować, usuwać i dodawać za pomocą przycisku Add Component.

Hierarchy (lista hierarchiczna) – Jest to lista elementów umieszczonych na scenie, które zachowują odpowiednią hierarchię.

Console – tutaj jest dostęp do danych, które można umieścić w świecie gry. Są to modele, tekstury, dźwięki, skrypty i inne

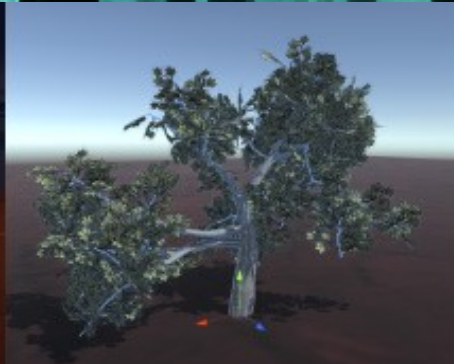
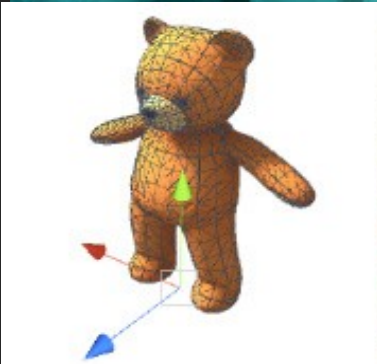
Podgląd sceny (środek) – miejsce ustawienia pozycji, rotacji i skali obiektów w przestrzeni. Zakładka Scene pokazuje podgląd na scenę gry, natomiast Game, przedstawia obraz z perspektywy gracza.

Toolbar – Pasek narzędziowy. Są tam umieszczone najważniejsze narzędzia i pola dotyczące sceny.



Game Object

Są to podstawowe obiekty w Unity, które reprezentują postacie, scenerię i rekwizyty.



Asset Store

Asset store jest magazynem, który udostępnia pobranie grafik, tekstur i obiektów, które zostały stworzone przez użytkowników. Zasoby są dostępne za darmo oraz odpłatnie. Następnie można je zaimportować do własnego projektu.

Scene Asset Store Game

CYBER WEEK MEGA SALE • OVER 2000 ASSETS UP TO 50% OFF

unity Asset Store

Search for assets

☐ On Sale Only (2506)

All Categories

- ☐ 3D (31391)
- ☐ 2D (7115)
- ☐ Add-Ons (38)
- ☐ Audio (5358)
- ☐ Essentials (60)
- ☐ Templates (2634)

Scene Asset Store Game

CYBER WEEK MEGA SALE • OVER 2000 ASSETS UP TO 50% OFF

unity Asset Store

Search for assets

Recommendations

See more

Release Date +

Have feedback on these filters?
[Tell us](#)



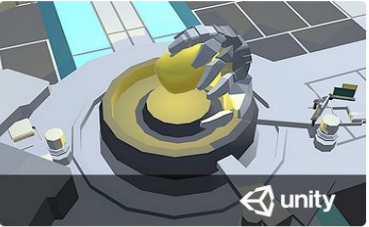
-50%

ASSET STORE ORIGINALS

Snaps Prototype | Asian Reside...

(not enough ratings)

\$9.99 \$4.99



-50%

ASSET STORE ORIGINALS

Snaps Prototype | Buried Memo...

(not enough ratings)

\$9.99 \$4.99

Hierarchy

Create

Main Camera

Directional Light

rabbit

Scene

Shaded

2D

Project

Create

Favorites

- All Materials
- All Models
- All Prefabs
- All Scripts

Assets

- MrRabZ
 - models
 - Blender
 - Materials
 - crate_wood
 - rabbit**
 - wall
- Standard

C# - Podstawy

o Funkcje

- ❖ Awake
- ❖ Start
- ❖ Update
- ❖ FixedUpdate



Funkcje

Awake - Ta funkcja wykonuje się jako pierwsza od razu po tym, jak dany obiekt zostanie utworzony na scenie.

Start – Funkcja wywoływana dla wszystkich utworzonych obiektów na scenie. Jest wywoływana po tym, gdy wszystkie metody Awake() na scenie zakończą swoje działanie.

Update – Funkcja, która wywołuje się na każdą ramkę gry.

FixedUpdate – Funkcja, wykonująca się co stały skok czasu.

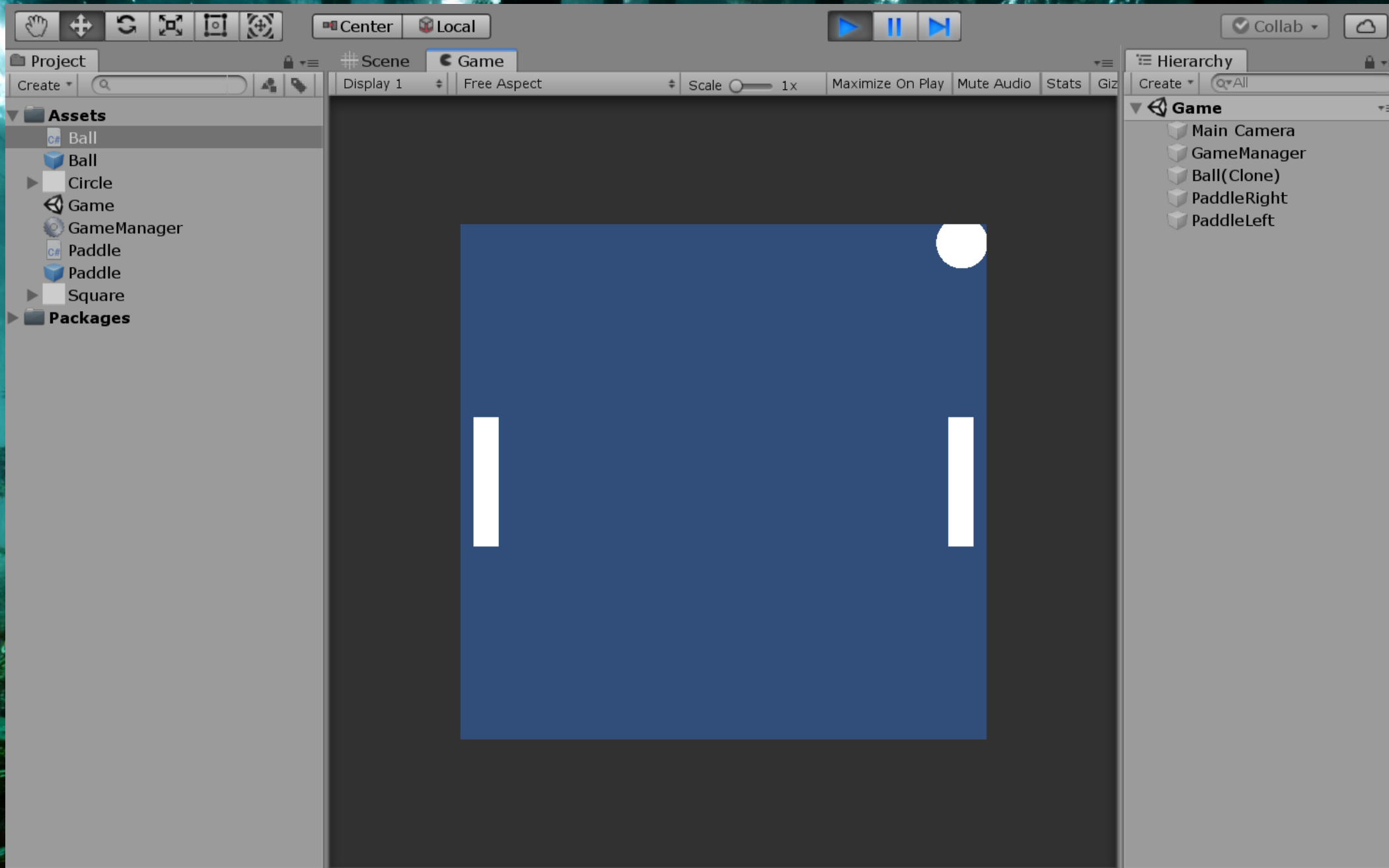
A misty forest scene with tall, slender trees and glowing yellow particles floating in the air. The scene is dimly lit, with light filtering through the trees.

Projektowanie własnej gry

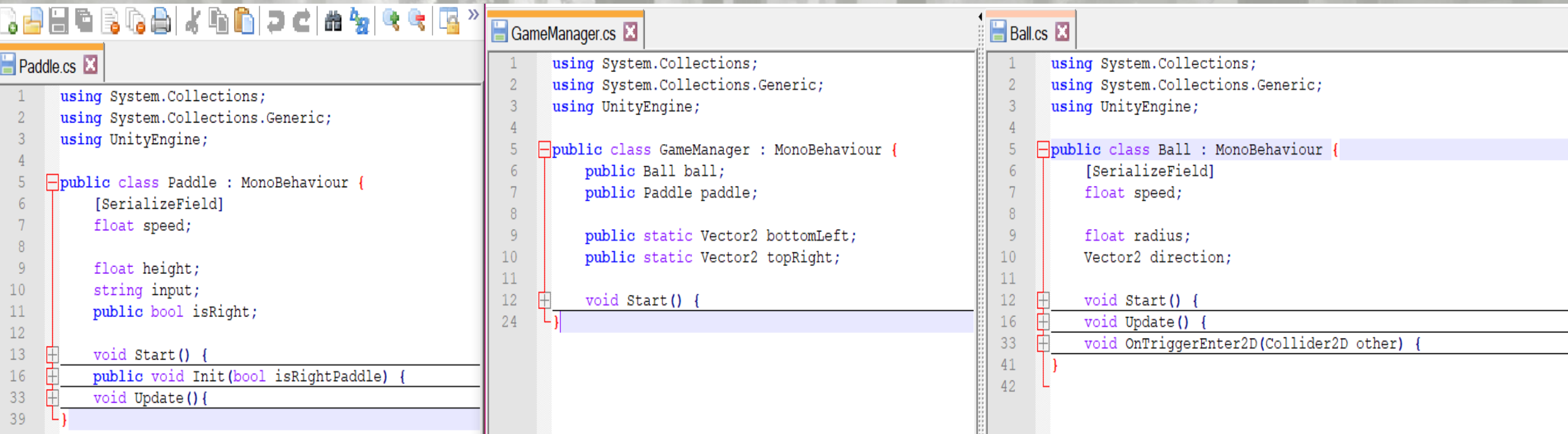
Przykład gry 2D: klasyczny Pong



Tworzenie programów w **Unity** odbywa się
poprzez interfejs graficzny jak
i zewnętrzne środowisko do tworzenia kodu.
Można wykorzystać IDE lub zwykły edytor tekstu.
Preferowane to **Visual Studio**.



Budowa mechaniki gry jest prosta, ale nie łatwo ją wymyśleć.

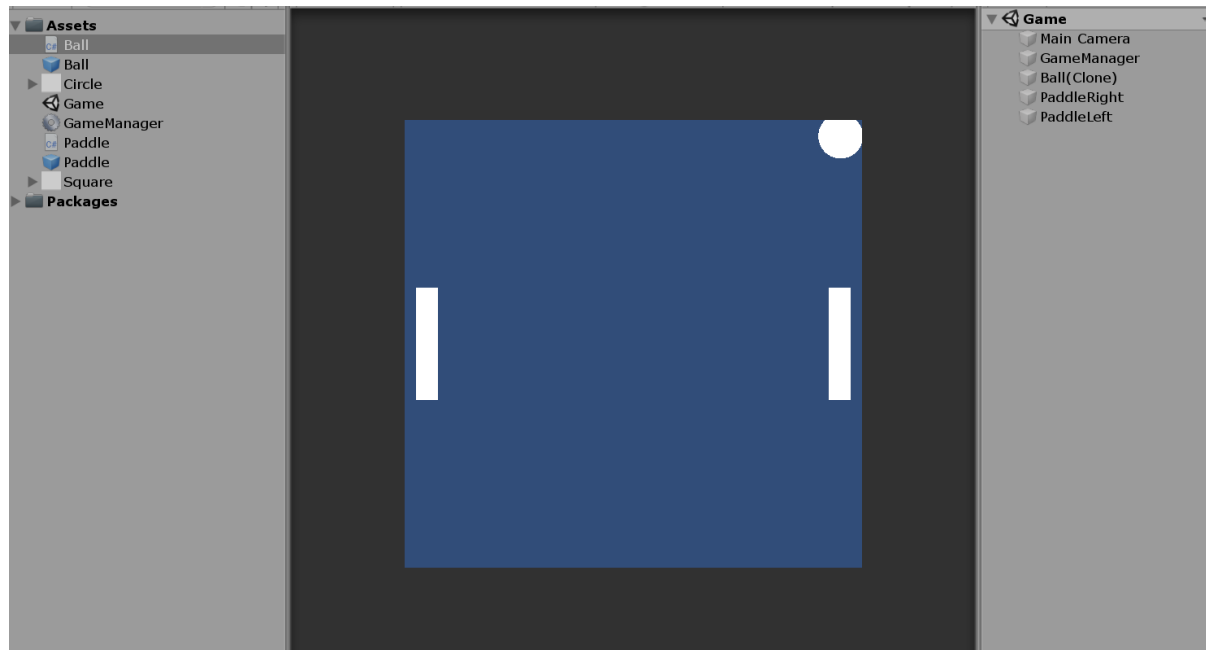


The image shows a screenshot of a Unity IDE with three C# scripts open: Paddle.cs, GameManager.cs, and Ball.cs. The scripts are written in C# and use UnityEngine namespaces.

```
Paddle.cs
1 using System.Collections;
2 using System.Collections.Generic;
3 using UnityEngine;
4
5 public class Paddle : MonoBehaviour {
6     [SerializeField]
7     float speed;
8
9     float height;
10    string input;
11    public bool isRight;
12
13    void Start() {
14    }
15    public void Init(bool isRightPaddle) {
16    }
17    void Update() {
18    }
19 }
```

```
GameManager.cs
1 using System.Collections;
2 using System.Collections.Generic;
3 using UnityEngine;
4
5 public class GameManager : MonoBehaviour {
6     public Ball ball;
7     public Paddle paddle;
8
9     public static Vector2 bottomLeft;
10    public static Vector2 topRight;
11
12    void Start() {
13    }
14 }
```

```
Ball.cs
1 using System.Collections;
2 using System.Collections.Generic;
3 using UnityEngine;
4
5 public class Ball : MonoBehaviour {
6     [SerializeField]
7     float speed;
8
9     float radius;
10    Vector2 direction;
11
12    void Start() {
13    }
14    void Update() {
15    }
16    void OnTriggerEnter2D(Collider2D other) {
17    }
18 }
```



```

1  using System.Collections;
2  using System.Collections.Generic;
3  using UnityEngine;
4
5  public class GameManager : MonoBehaviour {
6      public Ball ball;
7      public Paddle paddle;
8
9      public static Vector2 bottomLeft;
10     public static Vector2 topRight;
11
12     void Start() {
13         bottomLeft = Camera.main.ScreenToWorldPoint(new Vector2(0, 0));
14
15         topRight = Camera.main.ScreenToWorldPoint(new Vector2(Screen.width, Screen.height));
16
17         Instantiate(ball);
18
19         Paddle paddleRight = Instantiate(paddle) as Paddle;
20         Paddle paddleLeft = Instantiate(paddle) as Paddle;
21         paddleRight.Init(true);
22         paddleLeft.Init(false);
23     }
24 }

```



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

```
public class Paddle : MonoBehaviour {
    [SerializeField]
    float speed;

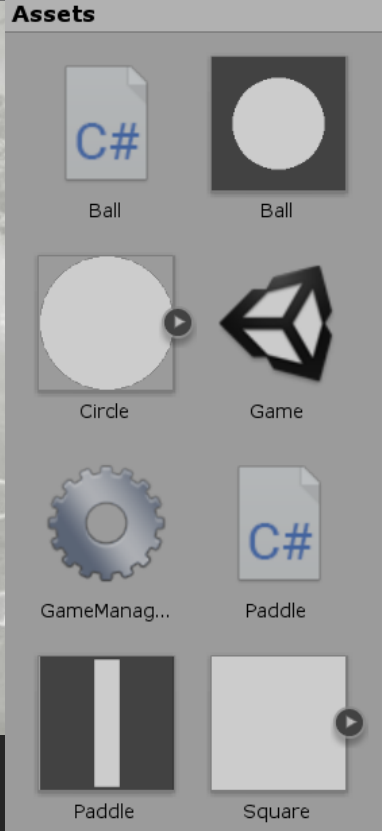
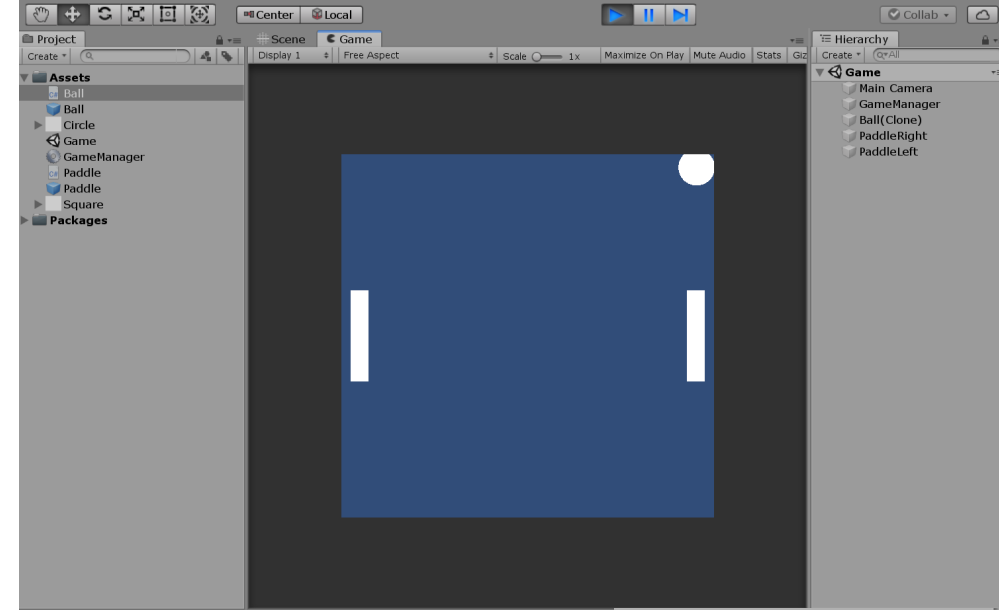
    float height;
    string input;
    public bool isRight;

    void Start() {
        height = transform.localScale.y;
    }

    public void Init(bool isRightPaddle) {
        isRight = isRightPaddle;
        Vector2 pos = Vector2.zero;

        if (isRightPaddle) {
            pos = new Vector2(GameManager.topRight.x, 0);
            pos -= Vector2.right * transform.localScale.x;
            input = "PaddleRight";
        }
        else {
            pos = new Vector2(GameManager.bottomLeft.x, 0);
            pos += Vector2.right * transform.localScale.x;
            input = "PaddleLeft";
        }
        transform.position = pos;
        transform.name = input;
    }

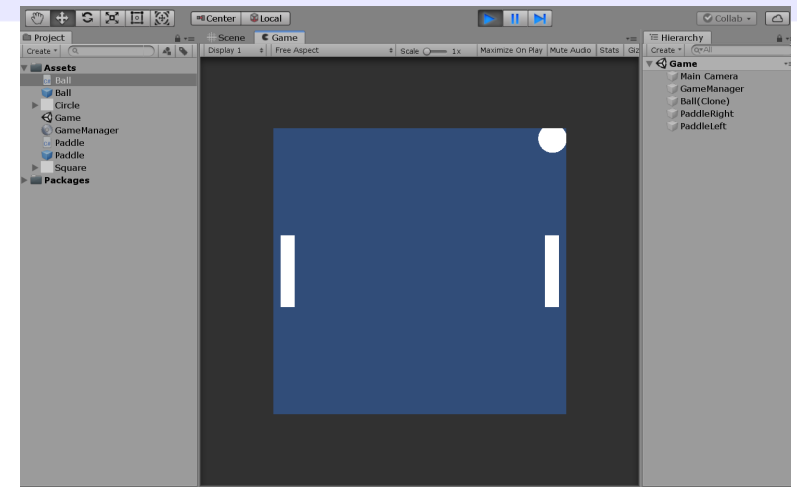
    void Update() {
        float move = Input.GetAxis(input) * Time.deltaTime * speed;
        if (transform.position.y < GameManager.bottomLeft.y + height / 2 && move < 0) { move = 0; }
        if (transform.position.y > GameManager.topRight.y - height / 2 && move > 0) { move = 0; }
        transform.Translate(move * Vector2.up);
    }
}
```

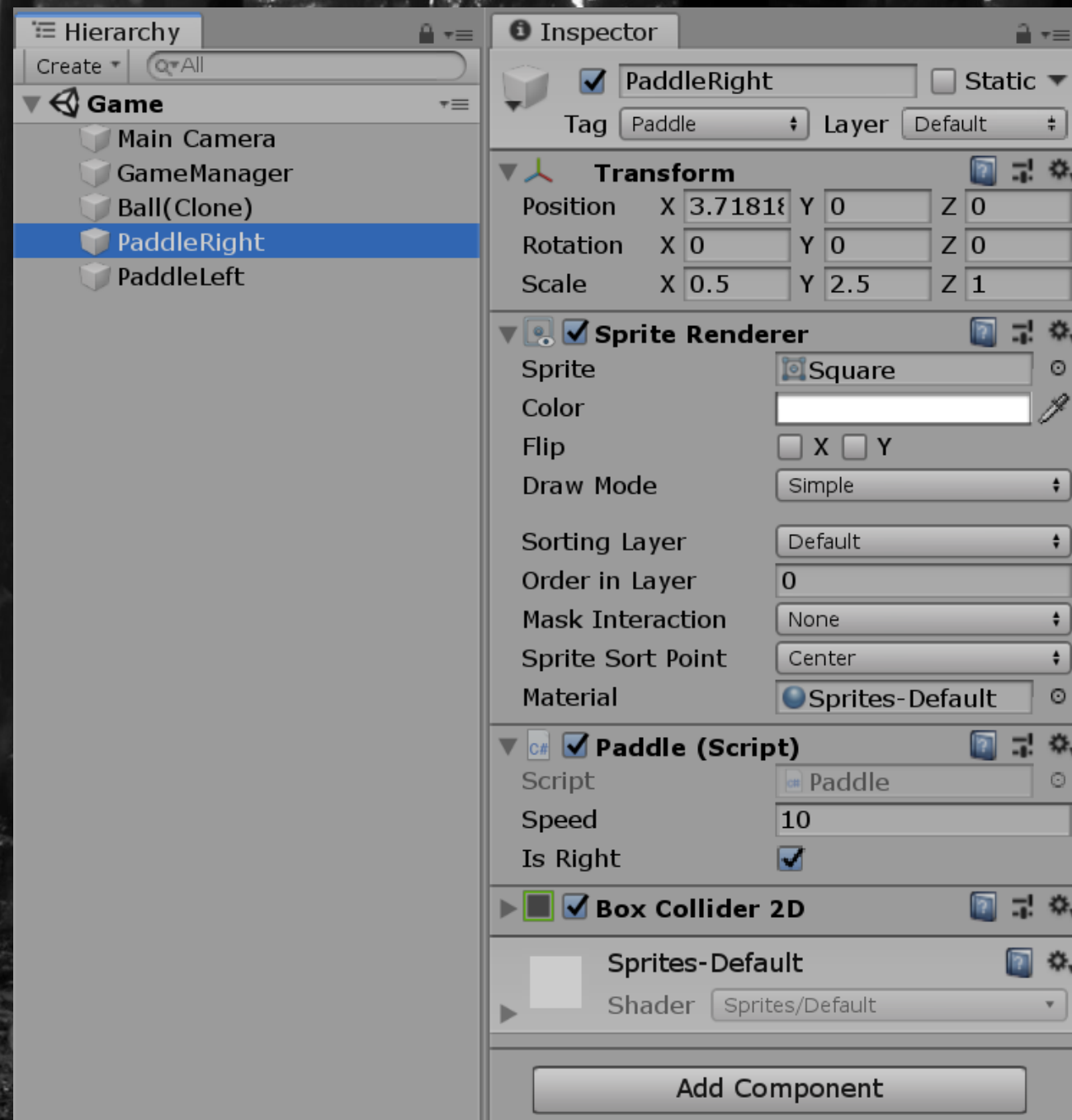
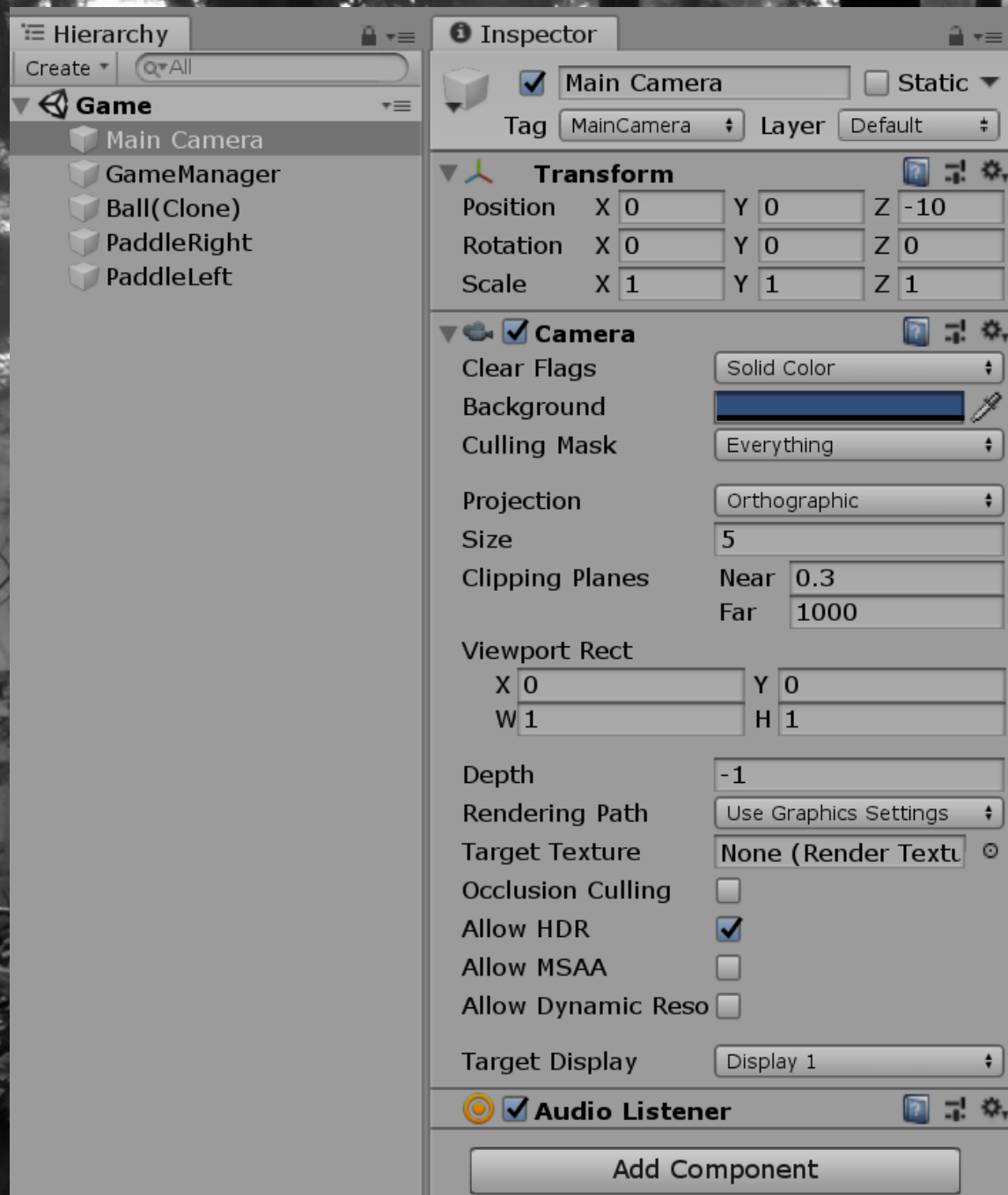


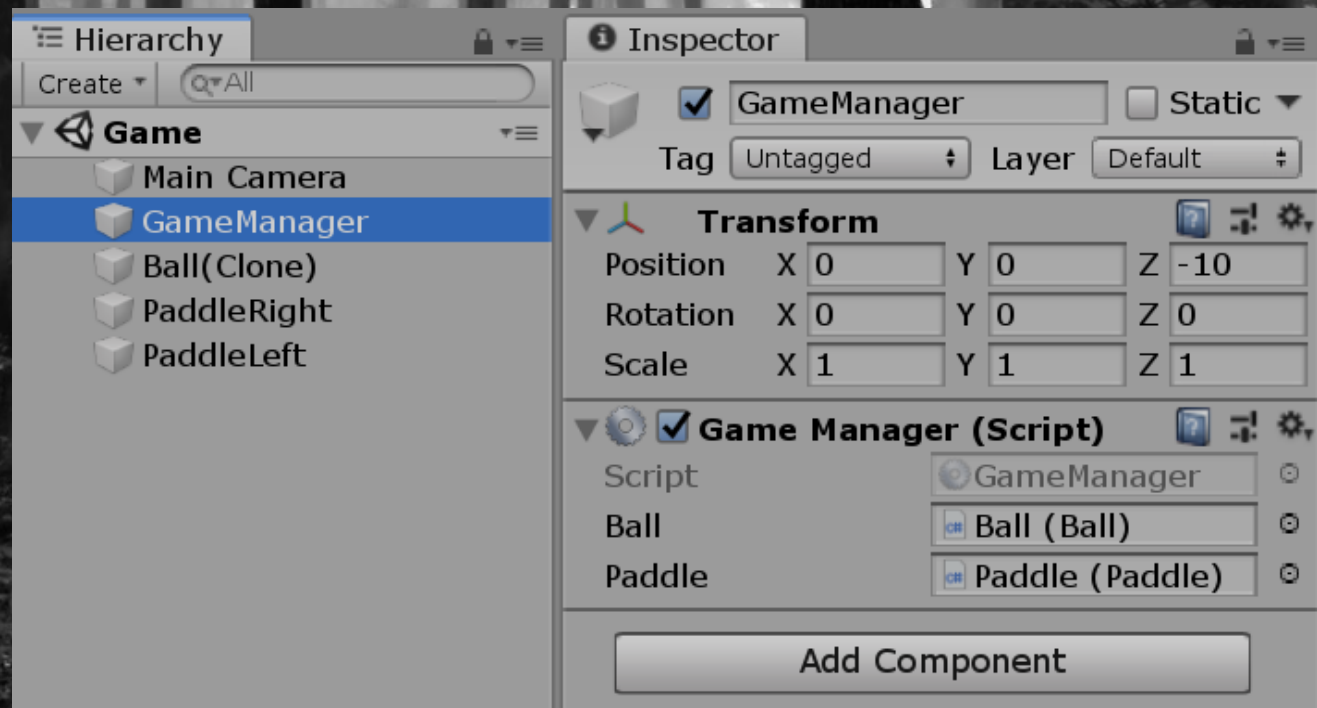
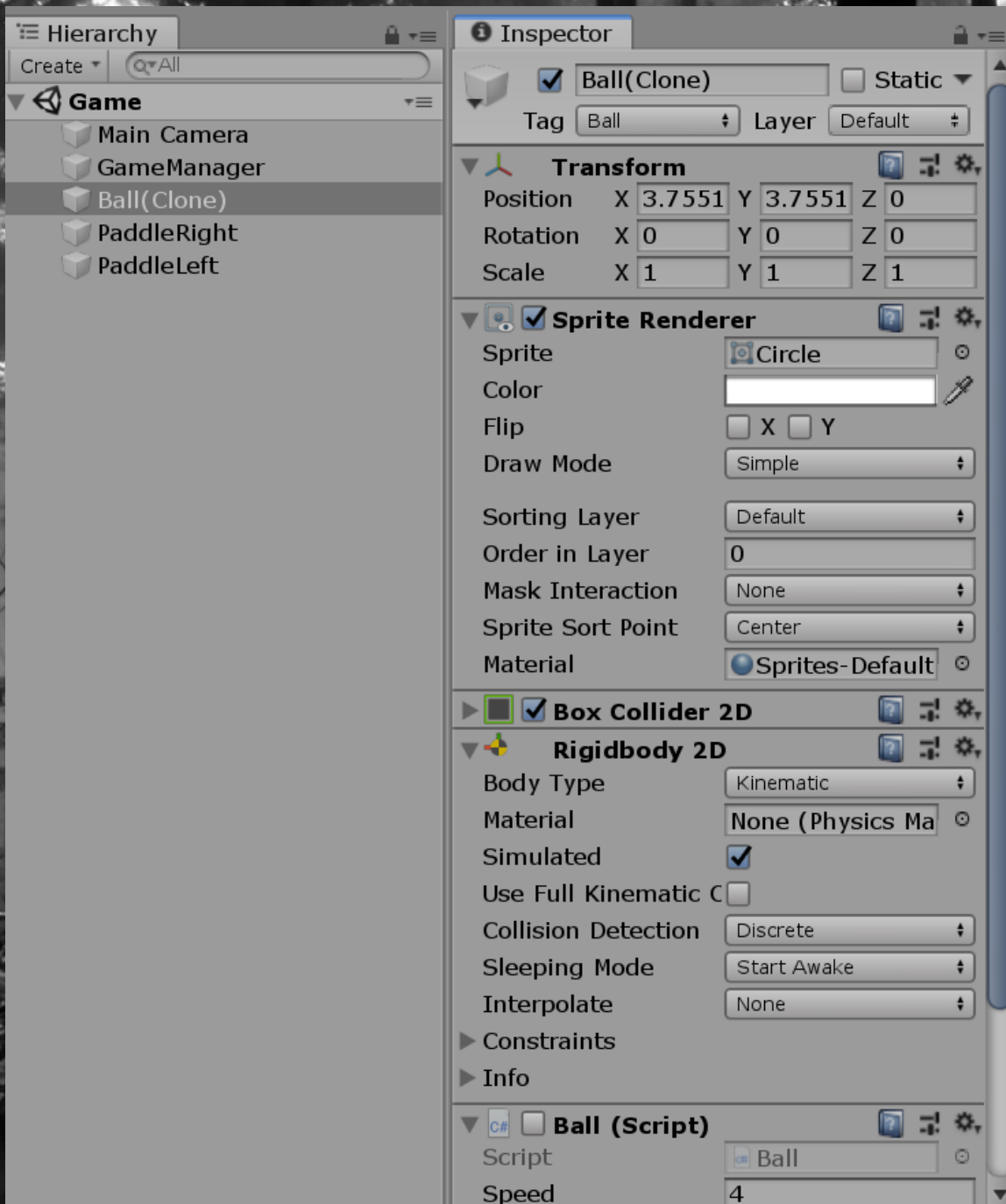
```

1 using System.Collections;
2 using System.Collections.Generic;
3 using UnityEngine;
4
5 public class Ball : MonoBehaviour {
6     [SerializeField]
7     float speed;
8
9     float radius;
10    Vector2 direction;
11
12    void Start() {
13        direction = Vector2.one.normalized;
14        radius = transform.localScale.x / 2;
15    }
16
17    void Update() {
18        transform.Translate (direction * speed * Time.deltaTime);
19        if (transform.position.y < GameManager.bottomLeft.y + radius && direction.y < 0) { direction.y = -direction.y; }
20        if (transform.position.y > GameManager.topRight.y - radius / 2 && direction.y > 0) { direction.y = -direction.y; }
21
22        // Game over
23        if (transform.position.x < GameManager.bottomLeft.x + radius && direction.x < 0) {
24            Debug.Log("Right player wins !!");
25            Time.timeScale = 0;
26            enabled = false;
27        }
28        if (transform.position.x > GameManager.topRight.x - radius && direction.x > 0) {
29            Debug.Log("Left player wins !!");
30            Time.timeScale = 0;
31            enabled = false;
32        }
33    }
34
35    void OnTriggerEnter2D(Collider2D other) {
36        if (other.tag == "Paddle") {
37            bool isRight = other.GetComponent<Paddle> ().isRight;
38
39            if(isRight == true && direction.x > 0) { direction.x = -direction.x; }
40            if(isRight == false && direction.x < 0) { direction.x = -direction.x; }
41        }
42    }

```







Znane gry stworzone w Unity

Unturned

Free to Play

