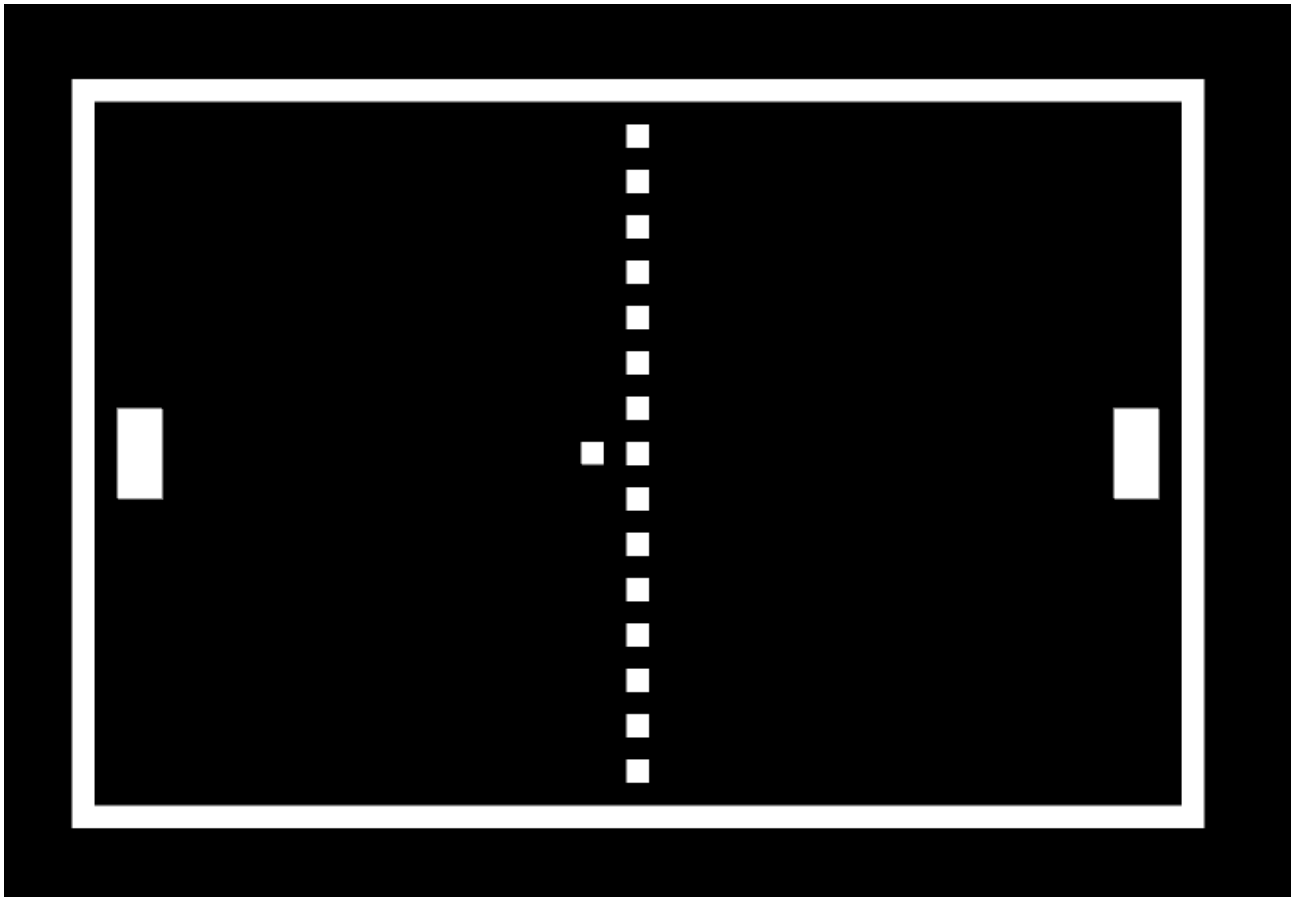
1. Prva laboratorijska vježba

U prvoj laboratorijskoj vježbi raditi će se jednostavna 2D igra - Pong. Cilj ove vježbe je upoznavanje s radom u Unity-u i osnove izrade 2D igara.

Pong (Atari Inc.) je jedna od prvih arkadnih video igara. Pong je dvodimenzionalna sportska igra koja simulira stolni tenis. Igrač u igri kontrolira reket pomičući ga okomito po lijevoj ili desnoj strani. Igrači koriste rekete kako bi odbijali lopticu naprijed natrag.

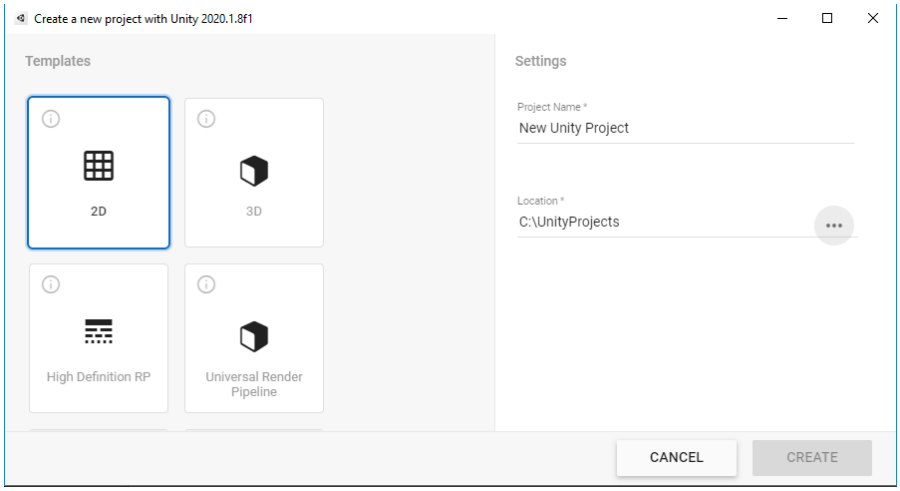


Slika 1 - Izgled gotove igre

Za ovu laboratorijsku vježbu je potrebno skinuti sa stranica kolegija .zip datoteku pod nazivom „PrvaLaboratorijskaVjezba“.

1.1. KREIRANJE NOVOG PROJEKTA

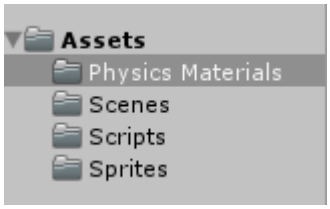
Potrebno je kreirati novi Unity 2D projekt kao što je prikazano na slici 2. Potrebno je odabrati kreiranje novog 2D projekta. Naziv projekta je proizvoljan.



Slika 2 - Kreiranje novog projekta

U novokreiranom projektu je potrebno unutar „Asset“ mape kreirati mape potrebne za organizaciju resursa potrebnih za kreiranje igre. Potrebne su mape:

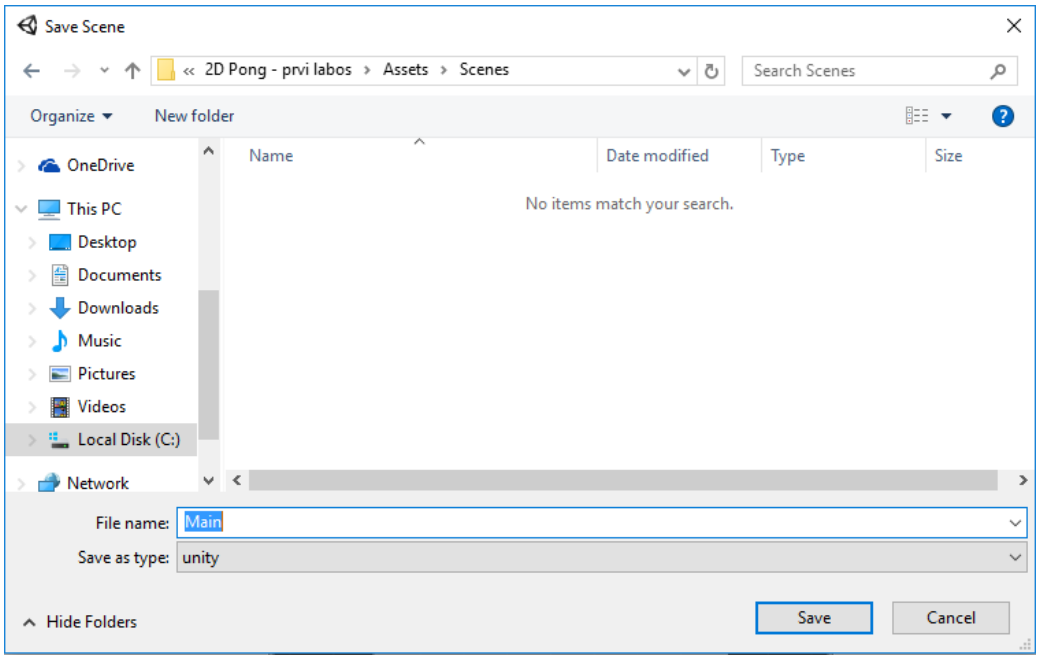
1. Physics Materials
2. Scenes
3. Scripts
4. Sprites



Slika 3 - Mape za organizaciju resursa

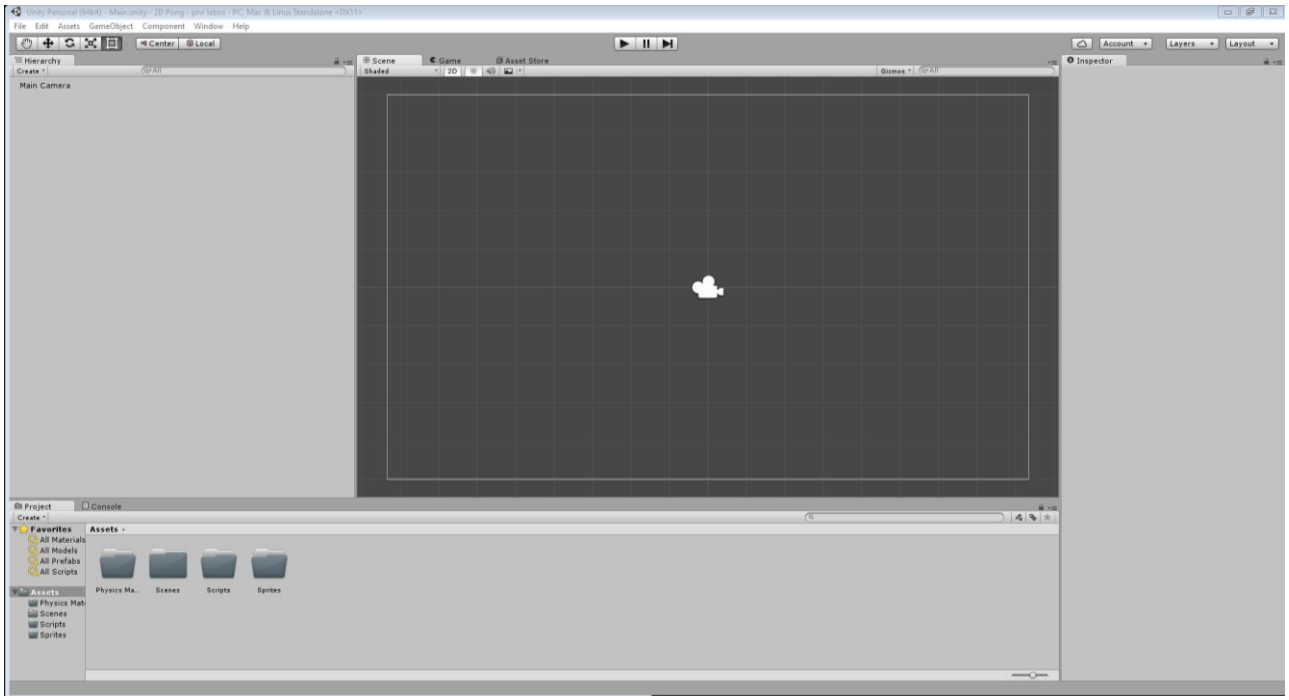
Skinuti sa stranica kolegija .zip datoteku pod nazivom „PrvaLaboratorijskaVjezba“ koja sadrži grafičke resurse za ovu laboratorijsku vježbu. Datoteku je potrebno raspakirati i slike prebaciti u „Sprites“ mapu projekta.

Potrebno je pohraniti scenu kako bi bilo moguće početi raditi igru. To je moguće na sljedeći način: „File -> Save Scene As…“. Scenu je potrebno pohraniti u „Scenes“ mapu pod nazivom „Main“.



Slika 4 - Pohrana nove scene

Izgled novokreiranog projekta:

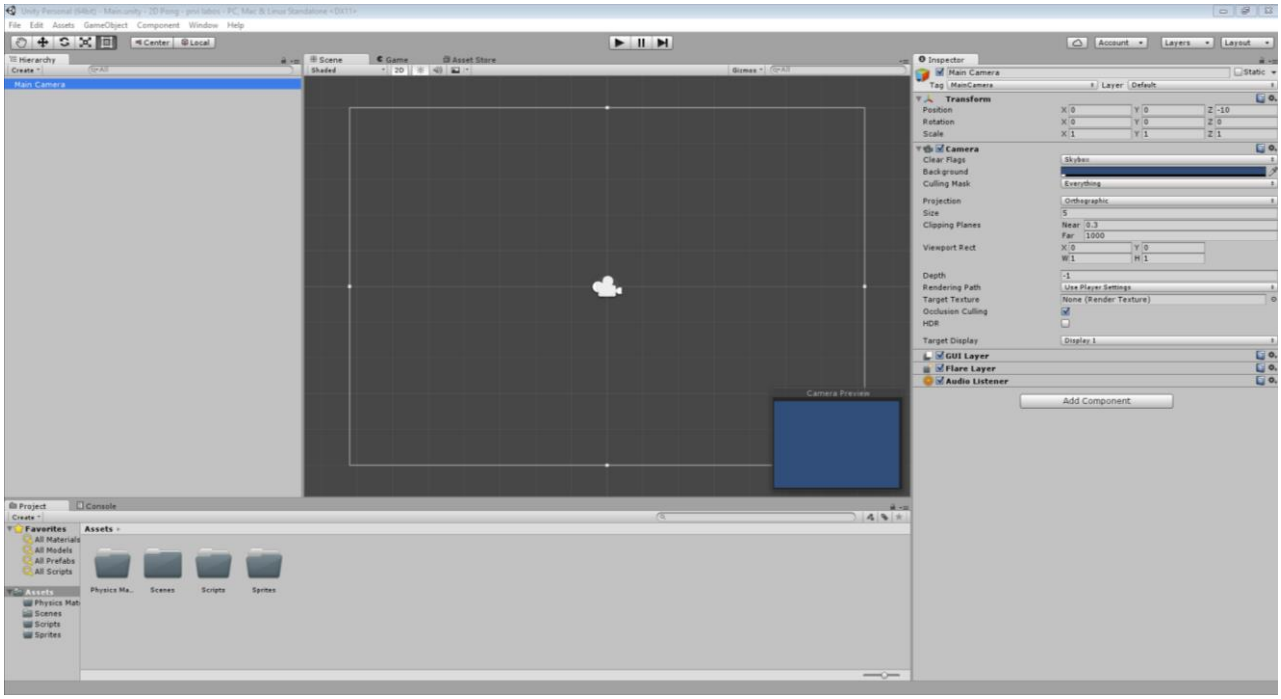


Slika 5 - Izgled novokreiranog projekta

1.2. NAMJEŠTANJE KAMERE I POZADINE

S obzirom na to da je želimo da je u pozadini samo crna boja, to je najlakše napraviti tako da se modificira glavna kamera („Main Camera“ objekt) koju je Unity sam kreirao.

Kada se objekt kamere selektira, unutar inspektora se mogu mijenjati njegove postavke i u sceni se pojavljuje prozorčić unutar kojeg se može vidjeti izgled same scene gledane kroz kameru. Tako će izgledati igra igraču kada ju pokrene.

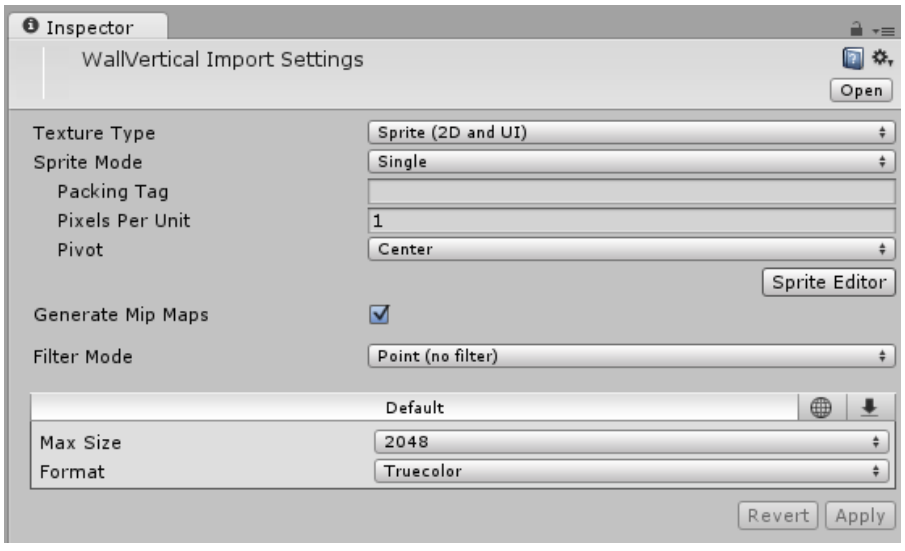


Slika 6 – Prikaz izgleda igre („Camera Preview“)

Promjena pozadinske boje igre se radi na način da se promijeni „Background“ postavka glavne kamere. Za potrebe ove igre je najbolje odabrati crnu pozadinsku boju.

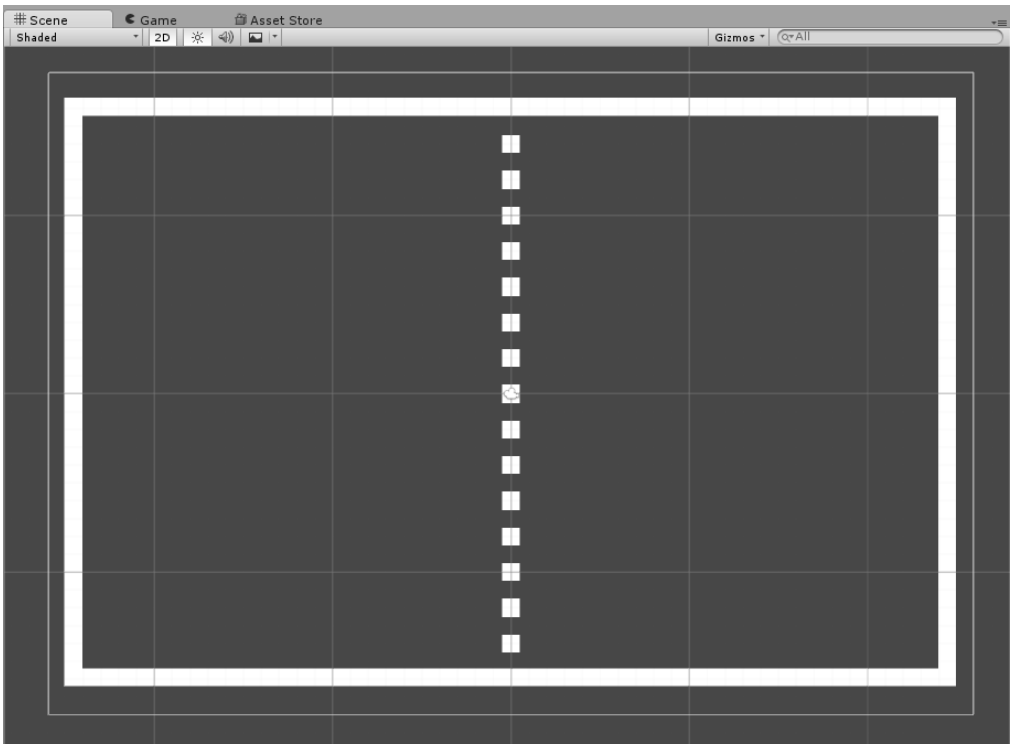
1.3. POSTAVLJANJE IZGLEDA SCENE

Kako bi se sve slike dobro prikazivale unutar scene, potrebno je složiti dobro postavke kao na slici 7.



Slika 7 - Postavke *Sprite*-a

U scenu je potrebno postaviti vanjske „zidove“ i središnju crtu tako da scena izgleda kao što je prikazano na slici 8.

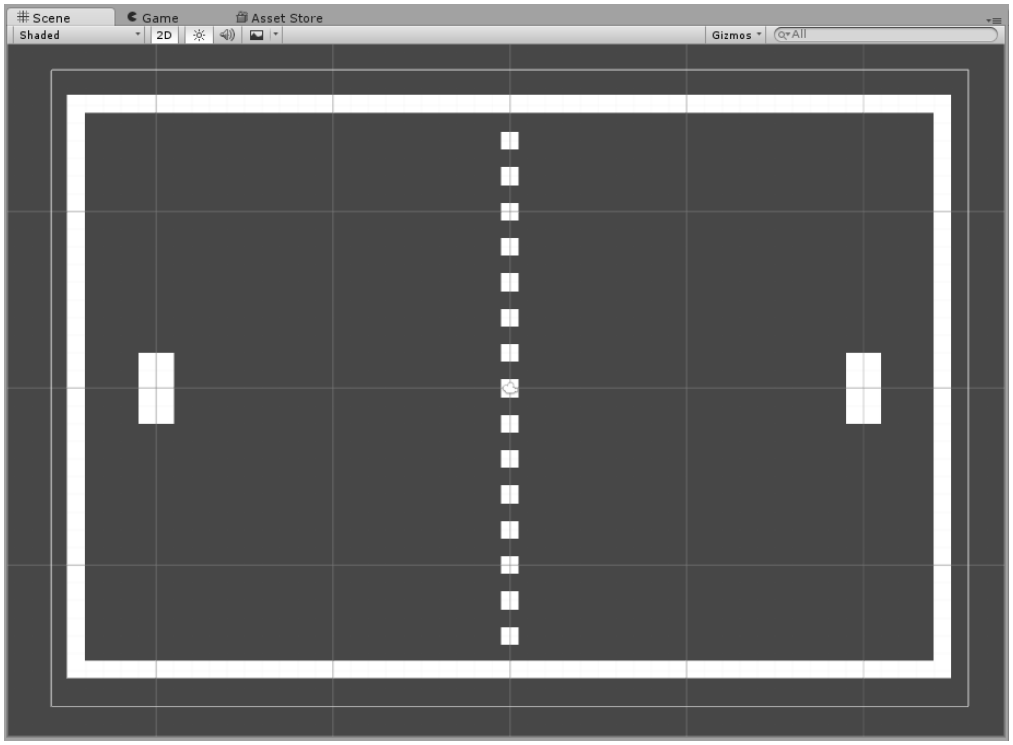


Slika 8 - Izgled scene

S obzirom na to da se želi da zidovi imaju fizičko prisustvo unutar scene kako loptica ne bi „izletjela“ van, potrebno im je dodati „Collider“ komponentu.

1.4. POSTAVLJANJE REKETA IGRAČA

Na jednak način kao i „zidove“ je potrebno dodati i rekete za igrače, po jednog na svakoj strani terena.



Slika 9 – Reketi igrača u sceni

1.4.1. Pokretanje reketa

Kako bi reketi imali fizičku prisutnost unutar same scene potrebno im je dodati „Collider“ komponentu. Također, kako bi ih mogli pomicati, potrebno im je dodati i „RigidBody“ komponentu.

Kako bi bilo moguće pomicati rekete, potrebno je kreirati novu C# skriptu. To je moguće napraviti: „Create -> C# script“. Novokreiranu skriptu je potrebno dodati u mapu „Scripts“ kako bi projekt bio lijepo posložen.

Primjer izgleda skripte za pomicanje reketa:

```
public class PlayerController : MonoBehaviour {

    // Brzina pomaka
    public float speed = 30;

    void FixedUpdate () {
        // Dohvat pomaka po vertikalnoj osi
        float vertical = Input.GetAxis("Vertical");

        // Pomicanje objekta
        GetComponent<Rigidbody2D>().velocity = new Vector2(0,
vertical) * speed;
```

```
        }
}
```

Ovu skriptu je potrebno dodati na oba reketa.

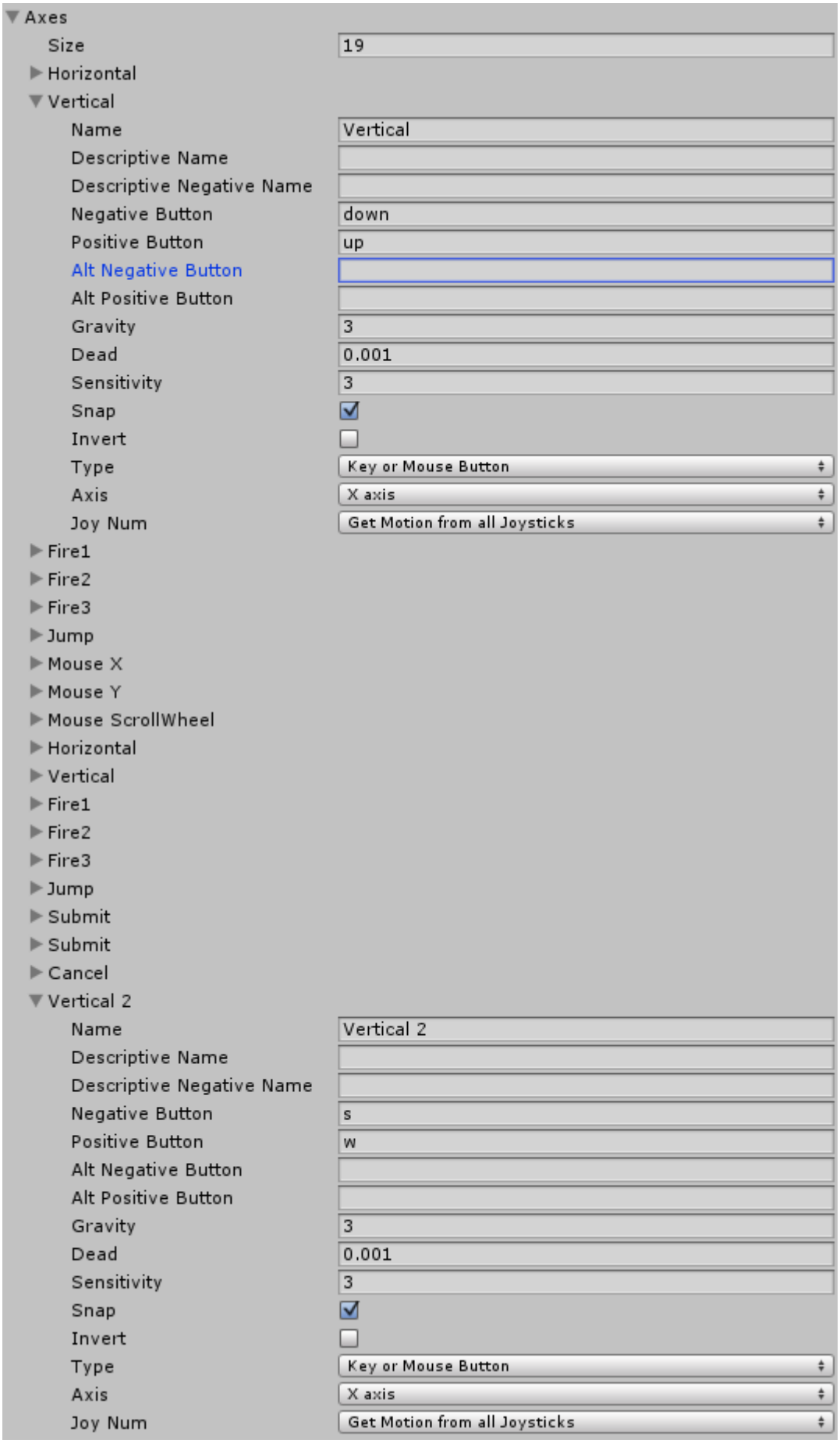
Ukoliko se igra u ovom trenutku pokrene, rekete će biti moguće pomicati pomoću strelica gore i dolje.

Igra se pokreče pritiskom na ikonicu „Play“.

Slika 10 - Pokretanje igre

S obzirom na to da se ne želi da se oba igrača pomiču jednako, potrebno je dodati još jedan set gumbiju za pokretanje. Za ovo je potrebno otvoriti „InputManager“ („Edit -> Project Settings -> Input“).

Potrebno je dodati novu os „Vertical 2“ i dodati novi set gumbiju pomoću kojih će se kontrolirati jedan od igrača.



Slika 11 - "Input Manager" sa dodanom "Vertical 2" osi

Kako bi svaki igrač upravljao jednim reketom, potrebno je modificirati skriptu za pomicanje tako da svaki od njih uzima kontrole od osi s drugim nazivom („Vertical“ i „Vertical 2“).

Napomena: naziv osi se piše kao string, a ako je varijabla public – moguće je mijenjati njenu vrijednost kroz sam Unity. Također je moguće proslijediti dvije različite vrijednosti iz dva različita objekta na poziv iste skripte.

1.5. DODAVANJE LOPTICE

Na isti način kako su dodani svi objekti do sada, potrebno je dodati i lopticu u scenu.

Kako bi se loptica pomicala potrebno joj je također dodati jednake komponente kao i za igrače.

Da bi se loptica odbijala od drugih objekata treba joj dodati materijal („Physics Material 2D“). Ovaj materijal ne bi trebao imati nikakvo trenje kako ne bi usporavao, te bi trebao imati maksimalnu vrijednost za odbijanje iz istog razloga.

Materijali se pohranjuju u mapu „Physics Materials“ a objektu se dodaje na njegovoj „Collider“ komponenti.

S obzirom na to da i u stvarnom svijetu loptica ima manju masu od reketa – potrebno joj je smanjiti vrijednost mase na „Rigidbody“ komponenti (na 0.0001).

Za pomicanje je također potrebno napraviti i C# skriptu koja će kontrolirati lopticu.

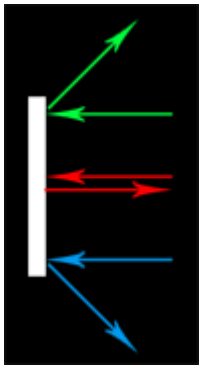
```
public class BallController : MonoBehaviour {


    public float speed = 30;


     void Start () {
        // Vector2.right daje vrijednost za označavanje desne
strane
        GetComponent<Rigidbody2D>().velocity = Vector2.right *
speed;
     }
}
```

U tu istu skriptu je potrebno dodati i dio programskog koda koji će provjeravati u kojeg igrača je loptica udarila i u koji dio igrača (reketa):

- Od lijevog reketa se loptica treba odbijati u desnu stranu
- Od desnog reketa se loptica treba odbijati u lijevu stranu
- Ako je loptica udarila u gornji dio reketa, treba se odbiti prema gore
- Ako je loptica udarila u srednji dio reketa, treba se odbiti ravno
- Ako je loptica udarila u donji dio reketa, treba se odbiti prema dolje



Slika 12 - Odbijanje loptice

Ovaj dio programskog koda izgleda na sljedeći način:

```
    void OnCollisionEnter2D(Collision2D col)
    {
        // Provjera da li je loptica udarila u lijevi reket
        if (col.gameObject.name == "RacketLeft")
        {
            // Provjera u koji dio reketa je loptica udarila
            float y = HitFactor(transform.position, col.transform.position,
col.collider.bounds.size.y);
            // Normalizacija vektora
            Vector2 direction = new Vector2(1, y).normalized;
            // Pomicanje loptice
            GetComponent<Rigidbody2D>().velocity = direction * speed;
        }

        // Provjera da li je loptica udarila u desni reket
        if (col.gameObject.name == "RacketRight")
        {
            // Provjera u koji dio reketa je loptica udarila
            float y = HitFactor(transform.position, col.transform.position,
col.collider.bounds.size.y);
            // Normalizacija vektora
            Vector2 direction = new Vector2(-1, y).normalized;
            // Pomicanje loptice
            GetComponent<Rigidbody2D>().velocity = direction * speed;
        }
```

```
    }


    private float HitFactor(Vector2 ballPosition, Vector2 racketPosition, float
racketHeight)
    {
        // 1 <- gornji dio reketa
        // 0 <- sredina reketa
        // -1 <- donji dio reketa
        return (ballPosition.y - racketPosition.y) / racketHeight;
    }
```

Također je potrebno dodati sve ostale potrebne postavke na objekte unutar igre kako bi ova skripta ispravno radila.

Ako se igra pokrene u ovom trenutku, loptica bi trebala letjeti i odbijati se od zidova i reketa, a svaki reket bi se trebao pomicati s drugačijim gumbima tipkovnice.