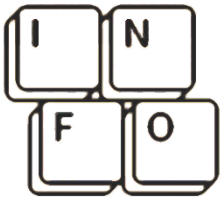


Colegiul Național de Informatică „Grigore Moisil”
Brașov

LUCRARE DE SPECIALITATE PENTRU OBȚINEREA ATESTATULUI PROFESIONAL

Profil matematică-informatică

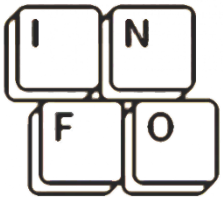
(intensiv informatică)

Candidat: Pașca Robert-Paul

Clasa: a XII – a E

Profesor coordonator: Penghiş Diana

2022



Colegiul Național de Informatică „Grigore Moisil”
Brașov

Tetris

Candidat: Pașca Robert-Paul

Clasa: a XII – a E

Profesor coordonator: Penghiș Diana

2022

Cuprins

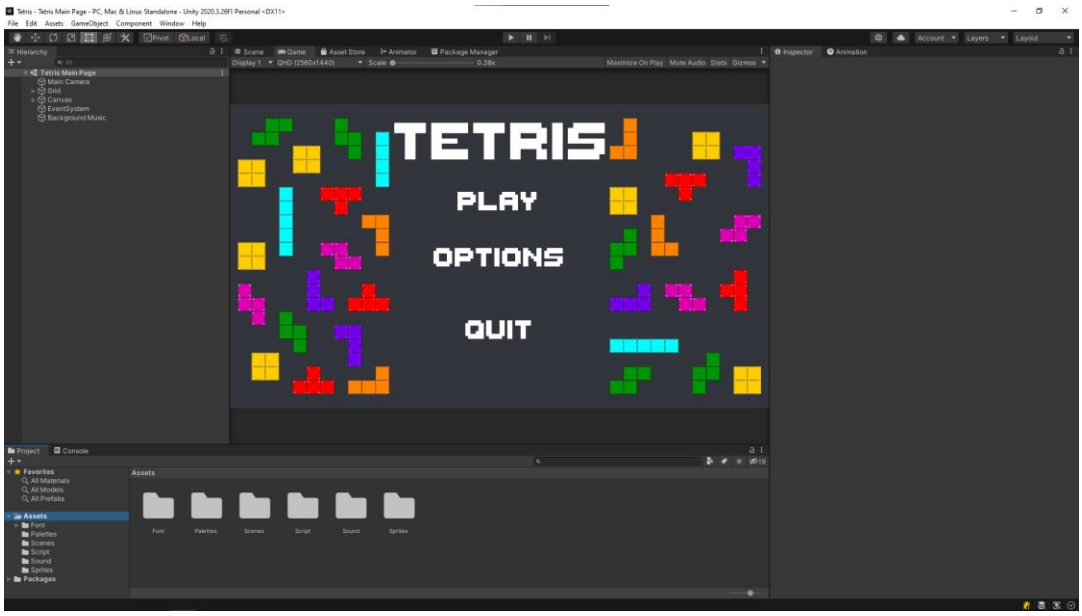
I. Motivația alegerii temei

Din dorința de a explora game engine-ul Unity m-am hotărât să fac un joc în acesta. Atunci când cumperi un joc de pe o platforma online, este ușor să criticam defectele şi eventualele bug-uri. Pentru a avea o mai bună înțelegere a procesului de crearea a unui joc şi eventualele dificultăți, consider ca Unity(game engine bazat pe limbajul de programare C#) este optim pentru început. Un aspect interesant este ca, deși jocul este 2D, acesta folosește toate cele trei axe de coordonate X, Y şi Z. Unity este conceput şi pentru jocurile 2D ce păstrează o perspectivă izometrică, în care toate elementele sunt într-un anumit unghi pentru a da aspect de adâncime.

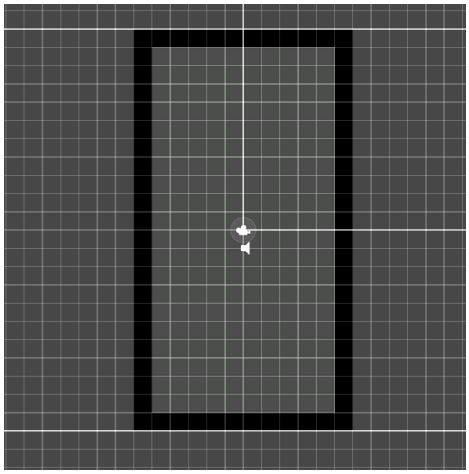
II. Descrierea aplicației

Jocul poate fi accesat relativ ușor, acesta putând fi rulat din fișierul Tetris.exe .

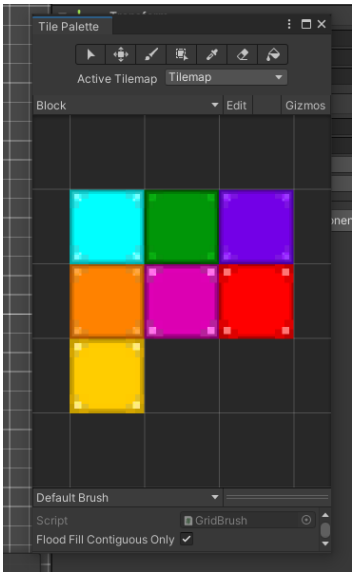
Jocul este creat folosind interfaţa Unity :



Acesta este făcut prin adăugarea unor sprite-uri (32x32 pixeli) ce împreună formează un Tilemap Rectangular. Sprite-urile sunt adăugate într-o paletă cu scopul de a putea fi puse pe grid. Tabla de joc este de dimensiune 10x20, aceasta respectând standardele Tetris.

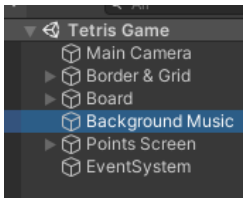


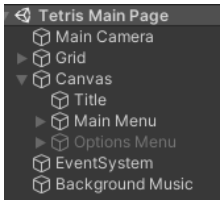
Poza 1 : Grid

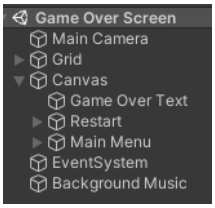


Poza 2 : Tile Palette

Jocul este structurat pe trei mari scene : Tetris Game , Tetris Main Page și Game Over Screen.





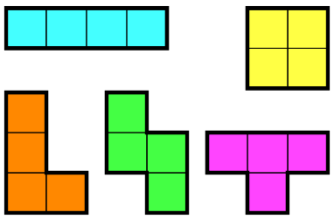


În “Tetris Game” se găsește “Border & Grid” ce reprezintă tabla de joc (Poza 1). Grid-ul “Board” alcătuiește corpul principal al jocului, acesta fiind locul în care se interconectează toate clasele şi funcţiile.

- Grid: componentă de tip canvas, în care se pot adăuga tilemap-uri
- Tilemap: obiectul în care se vor adauga tile-urile
- Tile Palette: asset ce definește o colecţie de tile-urile ce ar putea fi folosite
- Tile: asset ce poate conţine un Sprite, o culoare sau un colider

III. Detalii tehnice de implementare

A) Crearea pieselor tetromino



Un tetromino este o formă geometrică compusă din patru pătrate, conectate ortogonal (adică la margini și nu la colțuri). Tetromino-urile, ca și domino-urile și pentomino-urile, sunt un anumit tip de policub. Policubul corespunzător, numit tetracub, este o formă geometrică compusă din patru cuburi conectate ortogonal. Pentru a creea cele 7 piese am folosit un dicţionar ce retine fiecare poziţie a tile-ului în alcatuirea piesei. Iniţializarea acestora se realizează prin structura “TetrominoData” ce ia din dicţionar fiecare poziţie şi construieşte litera respectivă. Vector2Int este un tip de data ce reţine valorile a unor vectori în plan xOy.

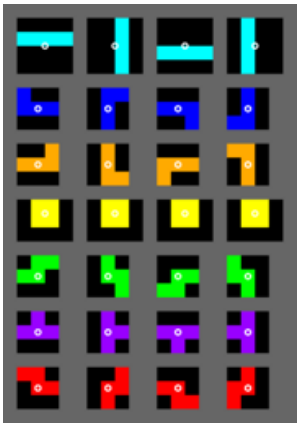
```
using UnityEngine.Tilemaps;
using UnityEngine;
21 references
public enum Tetromino
{
    I, J, L, O, S, T, Z
}
[System.Serializable]
4 references
public struct TetrominoData
{
    public Tile tile;
    public Tetromino tetromino;
    3 references
    public Vector2Int[] cells { get; private set; }
    4 references
    public Vector2Int[,] wallKicks { get; private set; }

    1 reference
    public void Initialize()
    {
        this.cells = Data.Cells[this.tetromino];
        this.wallKicks = Data.WallKicks[this.tetromino];
    }

}
```

```
public static readonly Dictionary<Tetromino, Vector2Int[]> Cells = new Dictionary<Tetromino, Vector2Int[]>()
{
    { Tetromino.I, new Vector2Int[] { new Vector2Int(-1, 1), new Vector2Int( 0, 1), new Vector2Int( 1, 1), new Vector2Int( 2, 1) } },
    { Tetromino.J, new Vector2Int[] { new Vector2Int(-1, 1), new Vector2Int(-1, 0), new Vector2Int( 0, 0), new Vector2Int( 1, 0) } },
    { Tetromino.L, new Vector2Int[] { new Vector2Int( 1, 1), new Vector2Int(-1, 0), new Vector2Int( 0, 0), new Vector2Int( 1, 0) } },
    { Tetromino.O, new Vector2Int[] { new Vector2Int( 0, 1), new Vector2Int( 1, 1), new Vector2Int( 0, 0), new Vector2Int( 1, 0) } },
    { Tetromino.S, new Vector2Int[] { new Vector2Int( 0, 1), new Vector2Int( 1, 1), new Vector2Int(-1, 0), new Vector2Int( 0, 0) } },
    { Tetromino.T, new Vector2Int[] { new Vector2Int( 0, 1), new Vector2Int(-1, 0), new Vector2Int( 0, 0), new Vector2Int( 1, 0) } },
    { Tetromino.Z, new Vector2Int[] { new Vector2Int(-1, 1), new Vector2Int( 0, 1), new Vector2Int( 0, 0), new Vector2Int( 1, 0) } },
};
```

B) Mutarea pieselor



Mutarea pieselor se realizează cu ajutorul sistemului “SRS”. Sistemul Super Rotation, cunoscut și sub denumirea de Standard Rotation System, este standardul actual al ghidului Tetris pentru modul în care se comportă tetromino-ele, definind unde și cum apar acestea, cum se rotesc și ce lovituri de perete pot efectua.

Orientările de initializare sunt incluse în poza din dreapta.

- Toate tetromino-urile apar orizontal și complet deasupra terenului de joc.
- Tetromino-ele I și O apar central, iar celelalte tetromino-e cu 3 celule lățime apar rotunjite la stânga.
- Tetromino-ele apar îndreptate în sus.

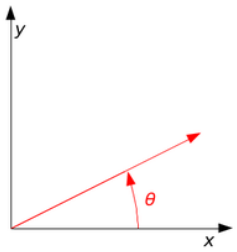
Rotirea pieselor se realizează pur matematic prin intermediul unei matrici de rotaţie. În algebra liniară, o matrice de rotație este o matrice de transformare care este utilizată pentru a efectua o rotație în spațiul euclidian.

$$R = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}$$

În două dimensiuni, matricea de rotație standard are următoarea formă : $R(\theta) = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}$.

Aceasta rotește vectorii coloană prin intermediul următoarei înmulțiri matrice. $\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$.

Astfel, noile coordonate (x′, y′) ale unui punct (x, y) după rotație sunt : $\begin{aligned} x' &= x\cos\theta - y\sin\theta \\ y' &= x\sin\theta + y\cos\theta \end{aligned}$.



O rotație în sens invers acelor de ceasornic a unui vector prin unghiul θ. Vectorul este inițial aliniat cu axa x.

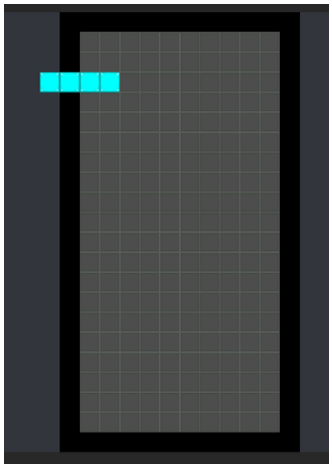
În codul jocului am creat funcţia Rotate cu parametrul direction care roteste piesa în sensul acelor de ceasornic, respectiv în sens invers al acelor de ceasornic. Dacă piesa de tetris se află lângă un zid funcţia verifică dacă acesta se poate roti, în caz contrar se întoarce la poziţia iniţială(bug-ul se poate observa în poza alaturată, unde în timpul uneia dintre cele 4 rotaţii piesa iese din câmpul jocului).

```
2 references
private void Rotate(int direction) /// 0 , 1 , 2 , 3
{
    int originalRotation = this.rotationIndex;
    this.rotationIndex = Wrap(this.rotationIndex + direction, 0, 4);

    RotationMatrix(direction);

    if (!TestWallKicks(this.rotationIndex, direction))
    {
        this.rotationIndex = originalRotation;
        RotationMatrix(-direction);
    }

}
```



Pentru a rezolva această problemă se foloseşte sistemul SRS care presupune o serie de 5 teste prin care se ajunge la concluzia dacă piesa se poate roti sau nu. De asemenea, acestea sunt adăugate în fişierul Data.cs sub forma unei liste.

Teste de coliziune pentru J , L , T , S , Z					
	Testul 1	Testul 2	Testul 3	Testul 4	Testul 5
0>>1	rotație de bază	(-1, 0)	(-1, 1)	(0,-2)[1]	(-1,-2)
1>>0	rotație de bază	(1, 0)	(1,-1)	(0, 2)	(1, 2)
1>>2	rotație de bază	(1, 0)	(1,-1)	(0, 2)	(1, 2)
2>>1	rotație de bază	(-1, 0)	(-1, 1)[1]	(0,-2)	(-1,-2)
2>>3	rotație de bază	(1, 0)	(1, 1)[1]	(0,-2)	(1,-2)
3>>2	rotație de bază	(-1, 0)	(-1,-1)	(0, 2)	(-1, 2)
3>>0	rotație de bază	(-1, 0)	(-1,-1)	(0, 2)	(-1, 2)
0>>3	rotație de bază	(1, 0)	(1, 1)	(0,-2)[1]	(1,-2)

Teste de coliziune pentru I					
	Testul 1	Testul 2	Testul 3	Testul 4	Testul 5
0>>1	rotație de bază	(-2, 0)	(1, 0)	(-2,-1)	(1, 2)
1>>0	rotație de bază	(1, 0)	(1,-1)	(0, 2)	(1, 2)
1>>2	rotație de bază	(2, 0)	(-1, 0)	(2, 1)	(-1,-2)
2>>1	rotație de bază	(-1, 0)	(2, 0)	(-1, 2)	(2,-1)
2>>3	rotație de bază	(2, 0)	(-1, 0)	(2, 1)	(-1,-2)
3>>2	rotație de bază	(-2, 0)	(1, 0)	(-2,-1)	(1, 2)
3>>0	rotație de bază	(1, 0)	(-2, 0)	(1,-2)	(-2, 1)
0>>3	rotație de bază	(-1, 0)	(2, 0)	(-1, 2)	(2,-1)

```
1 reference
private bool TestWallKicks(int rotationIndex, int rotationDirection)
{
    int wallKickIndex = GetWallKickIndex(rotationIndex, rotationDirection);

    for (int i = 0; i < this.data.wallKicks.GetLength(1); i++)
    {
        Vector2Int translation = this.data.wallKicks[wallKickIndex, i];

        if (Move(translation))
            return true;
    }
    return false;
}
```

```
2 references
private int Wrap(int input, int min, int max)
{
    if (input < min)
    {
        return max - (min - input) % (max - min);
    }
    else
    {
        return min + (input - min) % (max - min);
    }
}
```

Funcţia „Wrap” ne ajută în cazul rotirii piesei. Rotaţiile sunt salvate ca valori (0, 1, 2 şi 3). Funcţia ne ajută să revenim de la indicele 3 la indicele 0 , respectiv invers, fară a ieşi în afara limitelor precizate.

În ceea ce privește introducerea de la tastatură, Unity are deja funcții și clase predefinite care mapează tastele tastaturii. Functia HardDrop pastreaza miscarea din Tetris în care un Tetromino cade instantaneu, neputand fi mutat sau rotit ulterior.

```
1 reference
private void HardDrop()
{
    while (Move(Vector2Int.down))
        continue;
    Lock();
}
```

```
Unity Message | 0 references
private void Update()
{
    this.board.Clear(this);
    this.lockTime += Time.deltaTime;

    if (Input.GetKeyDown(KeyCode.Q))
    {
        Rotate(-1);
    }
    else if (Input.GetKeyUp(KeyCode.E))
    {
        Rotate(1);
    }
    if (Input.GetKeyDown(KeyCode.A))
    {
        Move(Vector2Int.left);
    }
    else if (Input.GetKeyDown(KeyCode.D))
    {
        Move(Vector2Int.right);
    }
    if (Input.GetKeyDown(KeyCode.S))
    {
        Move(Vector2Int.down);
    }
    if (Input.GetKeyDown(KeyCode.Space))
        HardDrop();

    if (Time.time >= this.stepTime)
        Step();


    this.board.Set(this);
}
```

IV. Cerințe hardware și software

Software:

- Windows 7, Windows 10 sau Windows 11, numai versiuni pe 64 de biți
- Drivere acceptate oficial de furnizorul de hardware

Hardware:

- CPU Intel Pentium 4 / AMD Athlon 64
- 2 GB Memorie RAM
- Plăci video capabile să ruleze DX10, DX11 şi DX12

V. Posibilitaţi de dezvoltare

Jocul deja prezintă standardele Tetris. În viitor acesta s-ar putea personaliza prin adăugarea unui nou nivel de dificultate cu piese diferite de cele standard.

VI. Concluzii

Am reuşit să dezvolt un joc modern şi cu standardele Tetris. Cred că elementele învăţate acum mă vor ajuta în luarea unei decizii pe viitor în legatură cu cariera mea.

VII. Bibliografie

Tetromino - Wikipedia

SRS | Tetris Wiki | Fandom

Rotation matrix - Wikipedia

https://youtu.be/zc8ac_qUXQY

https://youtu.be/ODLzYI4d-J8

https://youtu.be/T5P8ohdxDjo

https://assetstore.unity.com/packages/2d/fonts/free-pixel-font-thaleah-140059