

TWO-PLAYER TETRIS

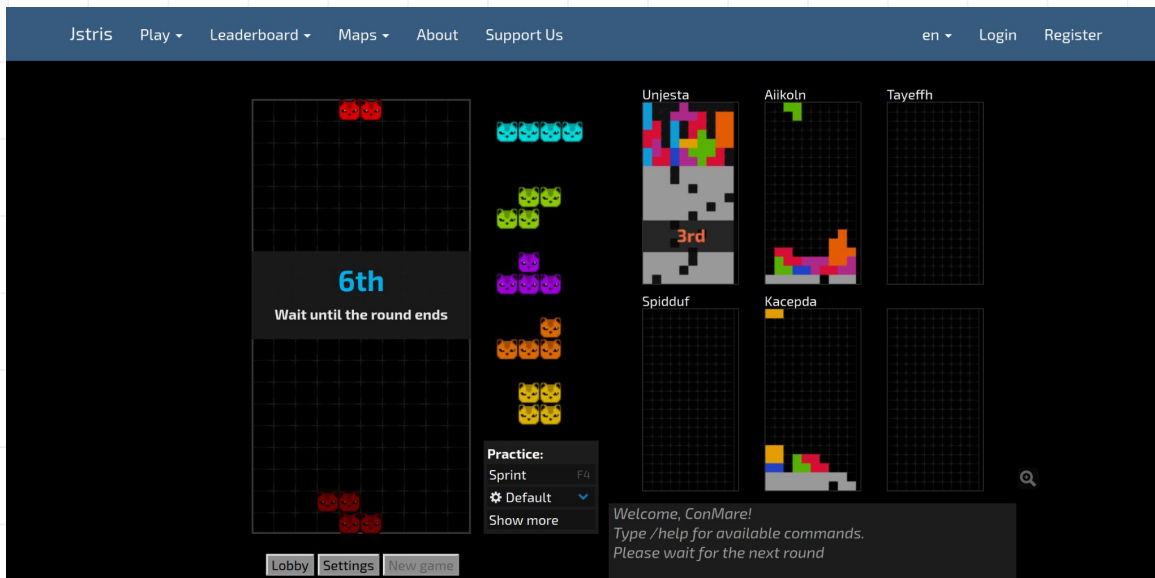
Michael Garcia & Daniel Garza

CIS 3365-01 / 04.25.2023



IDEA/CONCEPT

Original code was from <https://github.com/infantix/Tetris-2-players->





THEME

We used a retro/8-bit arcade theme, where we utilized code for a VHS filter effect:

<http://aleclownes.com/2017/02/01/crt-display.html>

Music was found on YouTube and combined in Audacity to make one long track:

<https://www.youtube.com/watch?v=dUpP80Y8YIQ>

<https://www.youtube.com/watch?v=q4tU7pmpOUk>

<https://www.youtube.com/watch?v=mhluBUa4mH4&t=33ls>

Images were found across Google, modified with GIMP:

https://www.freepik.com/premium-vector/keyboard-button-arrow-wasd-set-icon-simple-minimal-flat-vector-app-web-design_25481867.htm

<https://www.artstation.com/artwork/X1md4R>

<https://ar.pinterest.com/pin/593419688389266169/>

<http://www.rw-designer.com/cursor-set/retro-wave>

<https://www.stickpng.com/img/games/tetris/tetris-t-block>



JAVASCRIPT FUNCTIONS

Generates a random number which refers to a tetris piece case, created with another function for each piece.

```
let createLong = function () {  
  return [  
    [0, 2, 0, 0],  
    [0, 2, 0, 0],  
    [0, 2, 0, 0],  
    [0, 2, 0, 0]  
  ];  
}
```

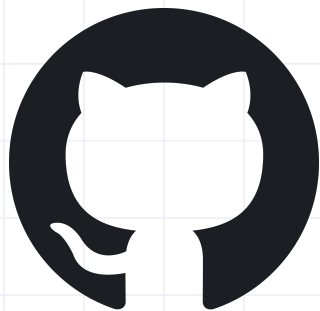
```
class PieceFactory  
{  
  createPiece() {  
    const numPieces = 7;  
    var pieceNum = Math.floor(Math.random() * numPieces) + 1;  
  
    switch (pieceNum) {  
  
      case 1:  
        return createSquare();  
  
      case 2:  
        return createLong();  
  
      case 3:  
        return createZigR();  
  
      case 4:  
        return createZigL();  
  
      case 5:  
        return createTri();  
  
      case 6:  
        return createLL();  
  
      case 7:  
        return createLR();  
    }  
  }  
}
```

JAVASCRIPT FUNCTIONS

Creates the controls for each player and associates them to the piece from the piece maker script. Also sets the score and speed/level for each player based on score.

```
increaseScore(num) {  
    let increase = 0;  
  
    if(num == 0) {  
        return;  
    }  
  
    if(num >= 1) {  
        increaseSpeed();  
        if(this.score >= this.level * 100) {  
            this.dropInterval -= (this.dropInterval / 2);  
            this.level++;  
        }  
    }  
}
```

```
class Player  
{  
    constructor(arena) {  
        this.arena = arena;  
        this.pieceFactory = new PieceFactory();  
  
        this.dropCounter = 0;  
        this.dropInterval = 1; //drop every second.  
        this.position = { x: 0, y: 0 };  
        this.matrix = [];  
        this.score = 0;  
        this.level = 1;  
  
        this.reset(); // init position and matrix.  
    }  
  
    moveLeft() {  
        this.position.x--;  
  
        if (this.arena.collide(this)) {  
            this.position.x++;  
        }  
    }  
  
    moveRight() {  
        this.position.x++;  
  
        if (this.arena.collide(this)) {  
            this.position.x--;  
        }  
    }  
}
```



LIVE DEMO

