

Projet-JS-Paris

Projet de Kevin

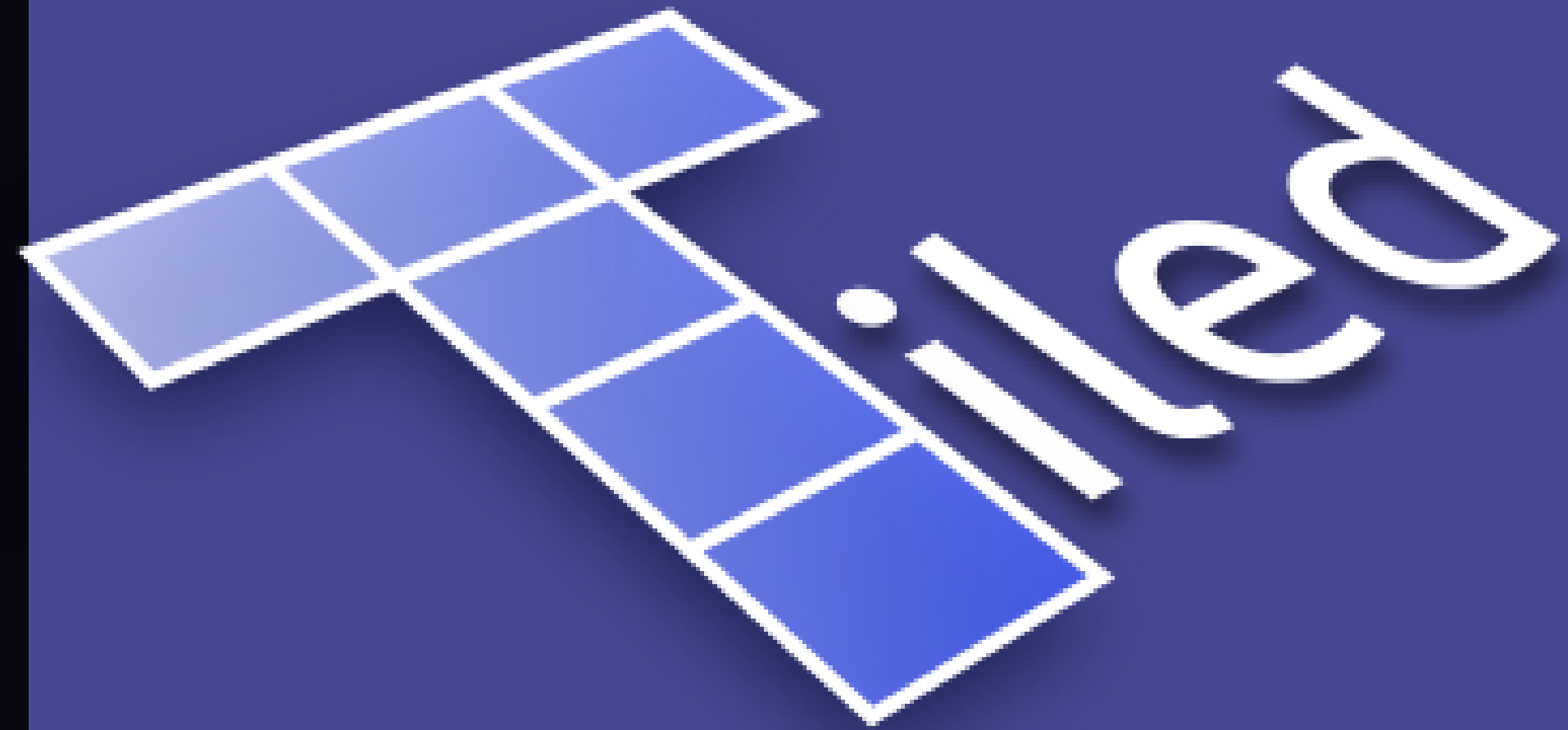
GE : Gather and Escape



présentation

Sommaire

- 1 - introduction/objectif
- 2 - ressources/assets
- 3 - Customisation maps
- 4 - Ressources Codes
- 5 - Collision
- 6 - Mouvements
- 7 - Coin
- 8 - Victoire
- 9 - Défaite
- 10 - Conclusion



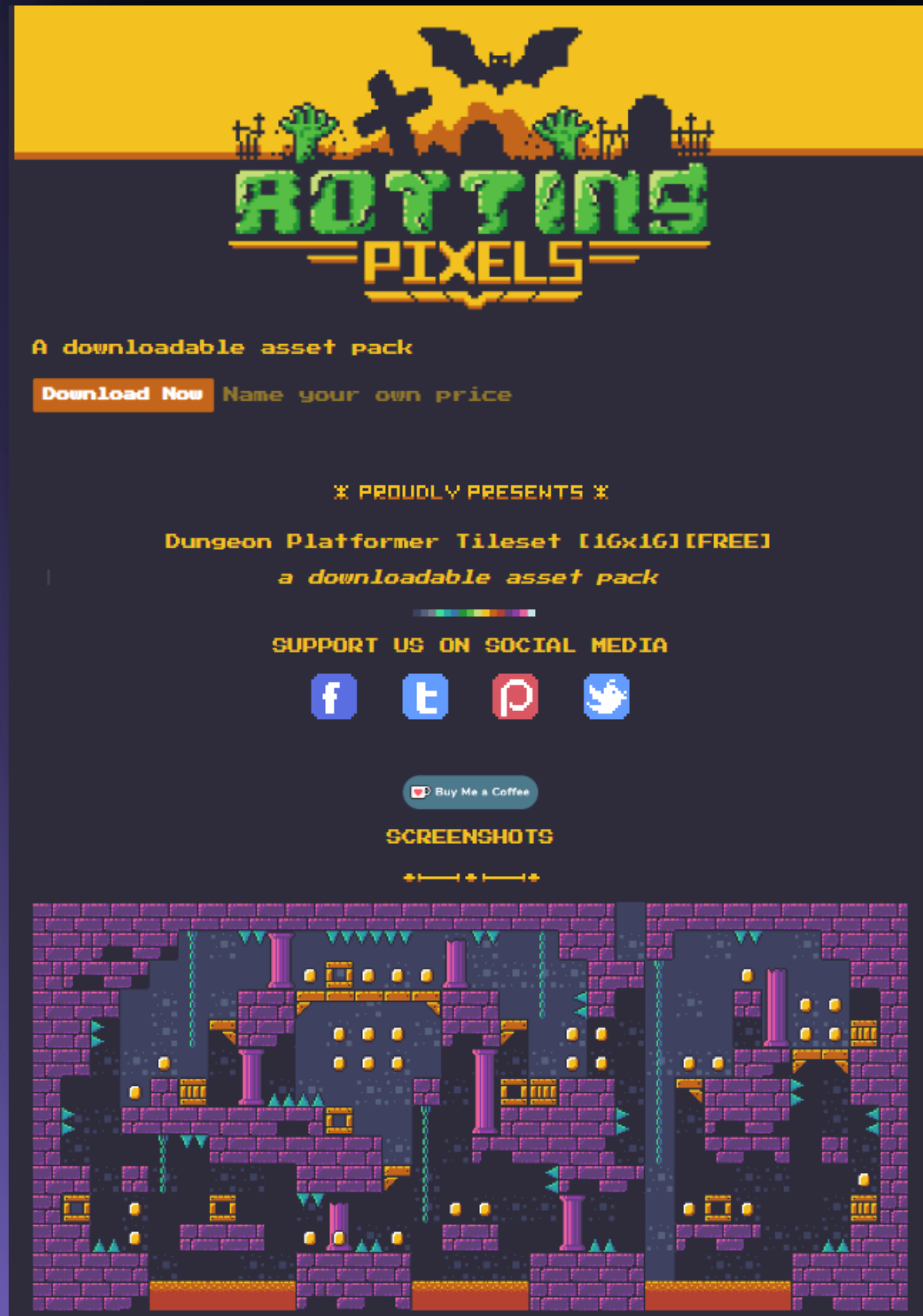
1 - Introduction/objectifs

Apprendre le javascript

Faire un jeu avec JS/HTML/CSS

Améliorer nos compétences de code

2 - Ressources Assets



Le joueur

La map



A screenshot of the 'Kings and Pigs' asset pack page. The page features a large, detailed screenshot of a game level made of brown and orange tiles. Below the screenshot, it says 'A downloadable asset pack' and 'Download Now Name your own price'. The title 'KINGS AND PIGS' is in large, red and green, pixelated letters. Below the title, there is a paragraph of text: 'The King Human came to recover his kingdom from the called "Green Skins". He has a big heavy hammer that can throw any Pig to the air, but is not gonna be so easy. Pigs are well organized guides by the one and only, The King Pig. They also have boxes and bombs to throw to you at any time, without mentioning that they also have cannons that spit FIREEEEE !!!'. Below the text, there is a section for 'Sprites:' with a list of items: King Human (10 animations), King Pig (8 animations), Pig (24 animations), Box, Bomb and Explosion, Cannon, Ball and Explosion, Diamond, Heart, Life and Diamond Bar, Dialogue boxes, Door, and Complete Tile-set (108 pieces). To the right of the text, there are three smaller screenshots of game levels made of brown and orange tiles.

A downloadable asset pack

Download Now Name your own price

KINGS AND PIGS

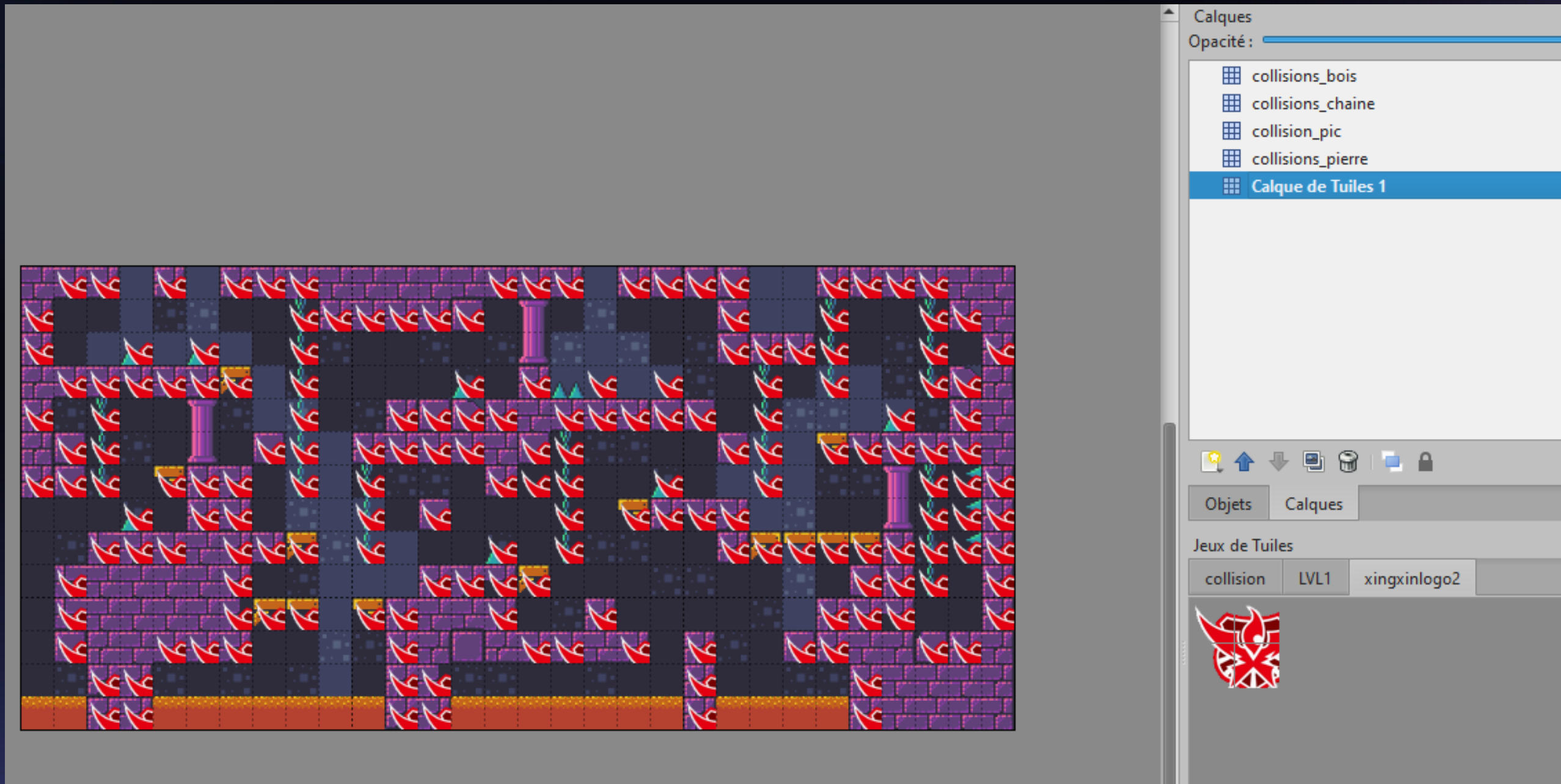
The King Human came to recover his kingdom from the called "Green Skins". He has a big heavy hammer that can throw any Pig to the air, but is not gonna be so easy. Pigs are well organized guides by the one and only, The King Pig. They also have boxes and bombs to throw to you at any time, without mentioning that they also have cannons that spit FIREEEEE !!!

Sprites:

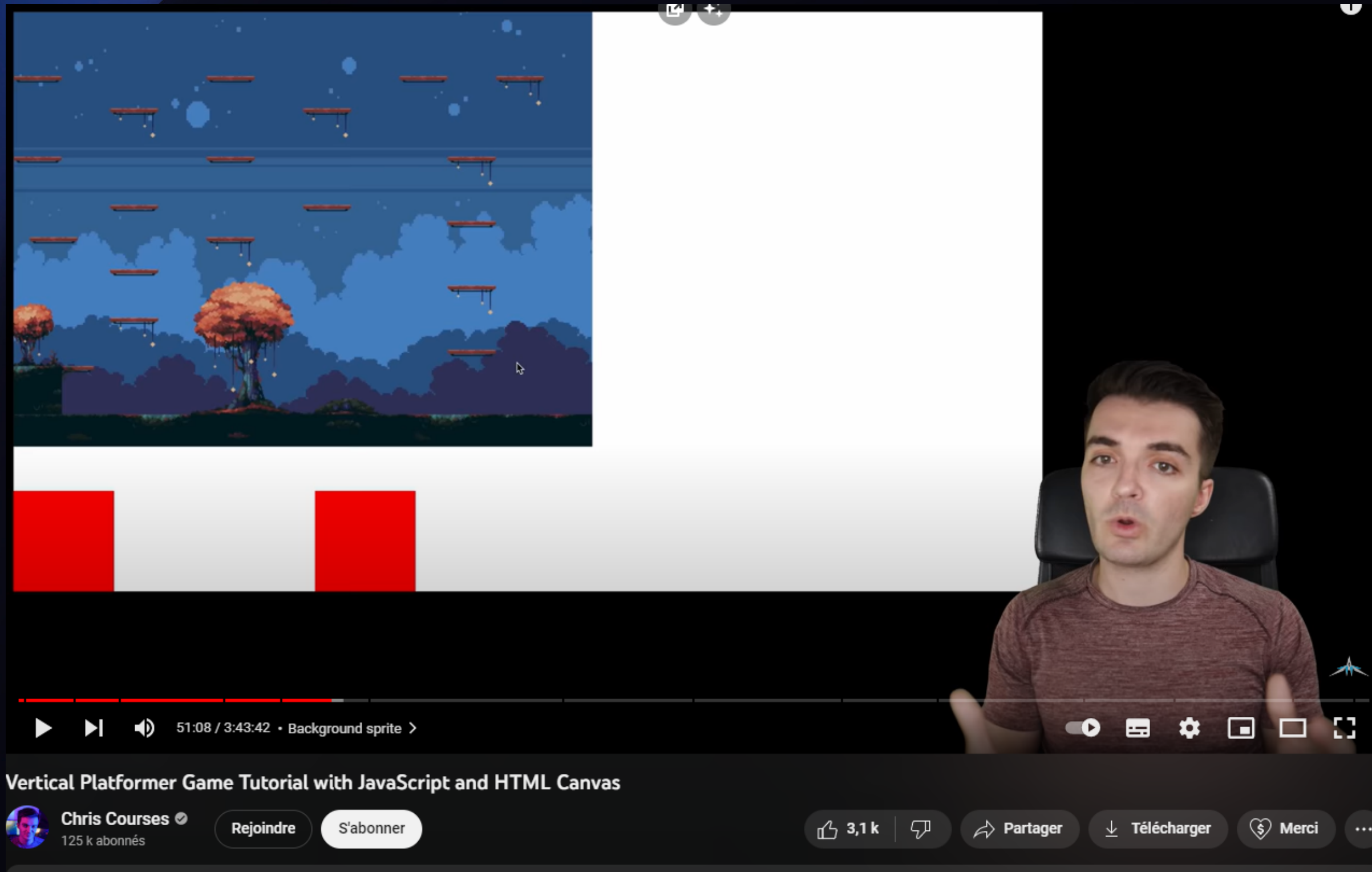
- King Human (10 animations)
- King Pig (8 animations)
- Pig (24 animations)
- Box
- Bomb and Explosion
- Cannon, Ball and Explosion
- Diamond
- Heart
- Life and Diamond Bar
- Dialogue boxes
- Door
- Complete Tile-set (108 pieces)

3 - Customisation maps

Tiled



4 - Ressources codes



The image shows a YouTube video player interface. The video content is split into two panels: the left panel displays a 2D platformer game scene with a dark blue night sky, floating wooden platforms, and a large orange tree on a dark ground; the right panel is a solid white rectangle. Below the video is a red progress bar. The video player controls at the bottom include a play button, a volume icon, a timestamp of 51:08 / 3:43:42, and a link to 'Background sprite >'. On the right side of the controls are icons for play/pause, full screen, settings, and other options. Below the video player, the video title 'Vertical Platformer Game Tutorial with JavaScript and HTML Canvas' is displayed. The channel name 'Chris Courses' is shown with a verified badge and '125 k abonnés'. There are two buttons: 'Rejoindre' and 'S'abonner'. To the right of these buttons are icons for likes (3,1 k), dislikes, share ('Partager'), download ('Télécharger'), and a heart icon with the text 'Merci'. A three-dot menu icon is also present.

Vertical Platformer Game Tutorial with JavaScript and HTML Canvas

Chris Courses ✓
125 k abonnés

Rejoindre S'abonner

3,1 k Partager Télécharger Merci ...

[illegible][illegible]

6 - Mouvements

```
index.html JS script.js M X JS keyBoard.js README.md M ...
JS script.js > play_game > animate > collisionBlocksWood.forEach() callback
407
408     if (keys.space.pressed && player.velocity.y === 0 && onFloor()) {
409         player.velocity.y = -3.2
410     }
411     for (let i = 0; i < collisionBlocksChain.length; i++) {
412         if (keys.z.pressed && climbing({object1: player.hitbox, object2: collisionBlocksChain[i]})) {
413             console.log('chain')
414             player.velocity.y = -0.5
415             player.lastDirection = 'Climb'
416             player.switchSprite('climb')
417         }
418     }
419     if (keys.q.pressed) {
420         player.lastDirection = 'Left'
421         player.switchSprite('runLeft')
422         player.velocity.x = -1.3
423         player.shouldPanCameraToTheRight({camera})
424     } else if (keys.d.pressed) {
425         player.lastDirection = 'Right'
426         player.switchSprite('runRight')
427         player.velocity.x = 1.3
428         player.shouldPanCameraToTheLeft({canvas, camera})
429     } else if (player.velocity.y === 0) {
430         if (player.lastDirection === 'Right') {
431             player.switchSprite('idleRight')
432         } else if (player.lastDirection === 'Left') {
433             player.switchSprite('idleLeft')
434         }
435     }
436     if (player.velocity.y < 0) {
437         player.shouldPanCameraDown({canvas, camera})
438         if (player.lastDirection === 'Right') {
439             player.switchSprite('jumpRight')
440         } else if (player.lastDirection === 'Left') {
441             player.switchSprite('jumpLeft')
442         }
443     } else if (player.velocity.y > 0) {
444         player.shouldPanCameraTop({canvas, camera})
445         if (player.lastDirection === 'Right') {
446             player.switchSprite('fallRight')
447         } else if (player.lastDirection === 'Left') {
448             player.switchSprite('fallLeft')
449         }
450     }
451     c.restore()
452 }
453 animate()
454 // setInterval(animate, 3000); //corresponding to 60 frame per second
455 }
456
457 function spikeDamage() {
458     this.life -= 0.1
459 }
```

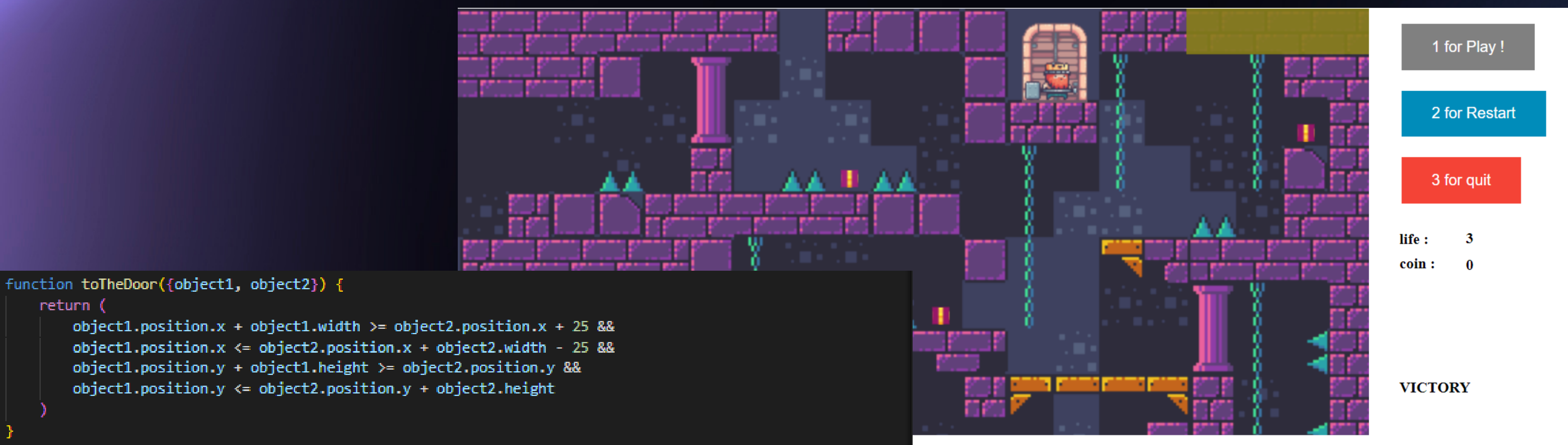
```
JS keyBoard.js > checkKeyBoard
1 function checkKeyBoard() {
2     window.addEventListener('keydown', (event) =>{
3         console.log(event)
4         switch (event.key) {
5             case ' ':
6                 keys.space.pressed = true
7         }
8         switch (event.key) {
9             case 'z':
10                 keys.z.pressed = true
11                 break
12             case 'q':
13                 keys.q.pressed = true
14                 break
15             case 'd':
16                 keys.d.pressed = true
17                 break
18             case 'Escape':
19                 keys.escape.pressed = true
20                 break
21             case 'a':
22                 keys.a.pressed = true
23                 break
24         }
25     })
26 })
27
28 window.addEventListener('keyup', (event) =>{
29     switch (event.key) {
30         case ' ':
31             keys.space.pressed = false
32         }
33     switch (event.key) {
34         case 'z':
35             keys.z.pressed = false
36             break
37         case 'q':
38             keys.q.pressed = false
39             break
40         case 'd':
41             keys.d.pressed = false
42             break
43         case 'Escape':
44             keys.escape.pressed = false
45             break
46         case 'a':
47             keys.a.pressed = false
48             break
49     }
50 })
51 }
```


7 - Coin

```
> index.html JS script.js M JS spriteCoin.js X README.md M
classes > JS spriteCoin.js > Coin > constructor
1 class Coin {
2   constructor({
3     coin,
4     position,
5     gathered,
6     imageSrc = './assets/Sprites/Coin.png',
7     frameRate = 4,
8     frameBuffer = 10,
9     scale = 1.5
10  }) {
11    this.coin = coin
12    this.position = position
13    this.gathered = gathered
14    this.scale = scale
15    this.image = new Image()
16    this.image.onload = () => {
17      this.width = (this.image.width / this.frameRate) * this.scale
18      this.height = this.image.height * this.scale
19      this.loaded = true
20    }
21    this.image.src = imageSrc
22    this.frameRate = frameRate
23    this.currentFrame = 0
24    this.frameBuffer = frameBuffer
25    this.elapsedFrames = 0
26    this.hitbox = {
27      position: {
28        x: this.position.x,
29        y: this.position.y,
30      },
31      width: 15,
32      height: 15,
33    }
34  }
35
36  draw() {
37    // hitbox
38    c.fillStyle = 'rgba(255, 0, 150, 0.5)'
39    c.fillRect(this.position.x, this.position.y, this.width, this.height)
40
41    if (!this.image) return
42    const cropbox = {
43      position: {
44        x: this.currentFrame * (this.image.width / this.frameRate),
45        y: 0,
46      },
47      width: this.image.width / this.frameRate,
48      height: this.image.height,
49    }
```

```
JS spriteCoin.js X
classes > JS spriteCoin.js > Coin > constructor
50   c.drawImage(
51     this.image,
52     cropbox.position.x,
53     cropbox.position.y,
54     cropbox.width,
55     cropbox.height,
56     this.position.x,
57     this.position.y,
58     this.width,
59     this.height,
60   )
61 }
62
63 update() {
64   this.updateHitBox()
65   this.draw()
66   this.updateFrames()
67 }
68
69 updateHitBox() {
70   this.hitbox = {
71     position: {
72       x: this.position.x,
73       y: this.position.y,
74     },
75     width: this.width,
76     height: this.height,
77   }
78 }
79
80 updateFrames() {
81   this.elapsedFrames++
82   if (this.elapsedFrames % this.frameBuffer === 0) {
83     if (this.currentFrame < this.frameRate - 1) this.currentFrame++
84     else this.currentFrame = 0
85   }
86 }
87 }
```

```
> index.html JS coins.js X JS script.js M README.md M
data > JS coins.js > ...
1 const coinsList = [
2   {coin: 'coin01', position: {x: 40, y: 80}, gathered: false},
3   {coin: 'coin02', position: {x: 907, y: 81}, gathered: false},
4   {coin: 'coin03', position: {x: 587, y: 113}, gathered: false},
5   {coin: 'coin04', position: {x: 40, y: 144}, gathered: false},
6   {coin: 'coin05', position: {x: 651, y: 209}, gathered: false},
7   {coin: 'coin06', position: {x: 40, y: 272}, gathered: false},
8   {coin: 'coin07', position: {x: 395, y: 272}, gathered: false},
9   {coin: 'coin08', position: {x: 262, y: 304}, gathered: false},
10  {coin: 'coin09', position: {x: 491, y: 337}, gathered: false},
11  {coin: 'coin10', position: {x: 907, y: 337}, gathered: false},
12 ]
13
14
```



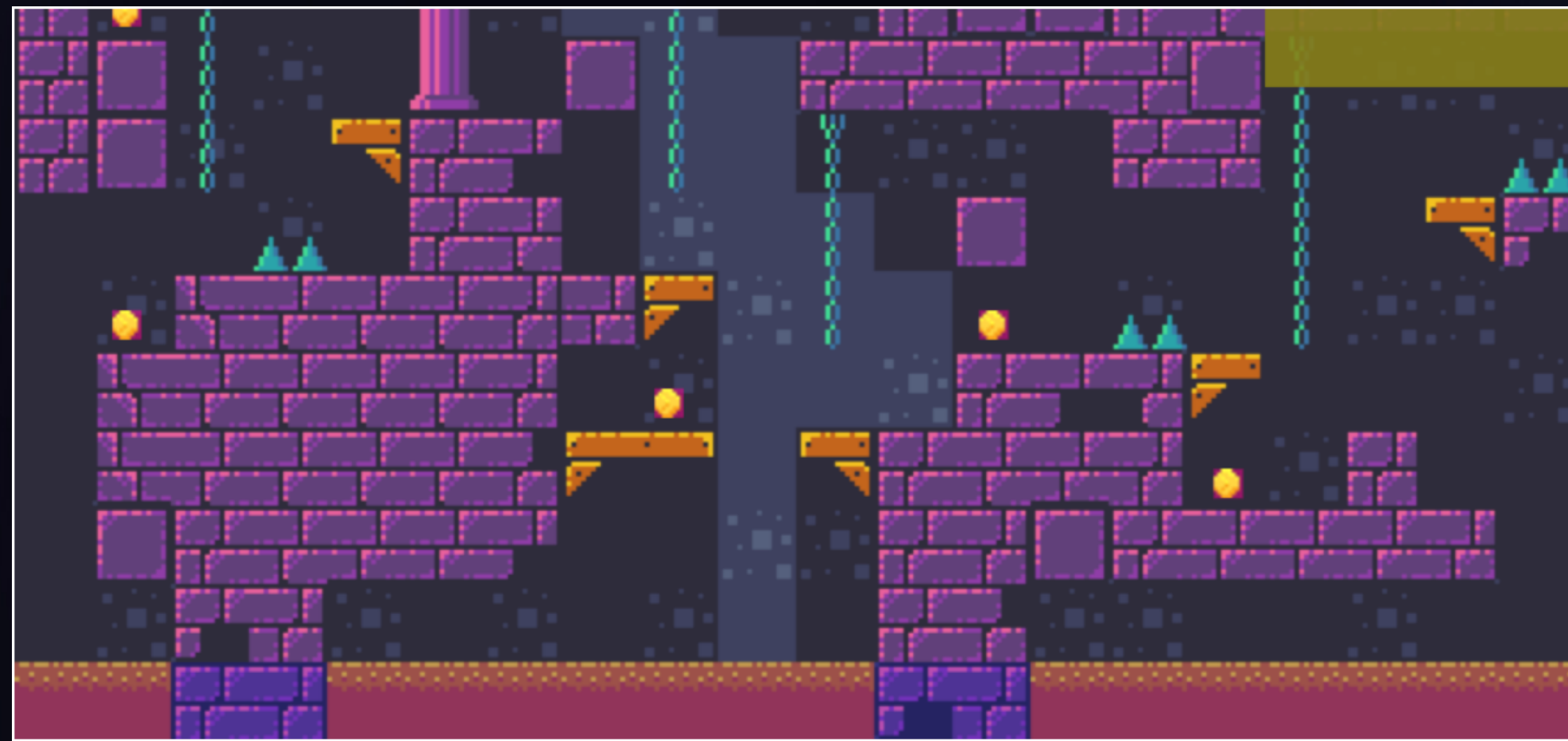
```
function toTheDoor({object1, object2}) {  
  return (  
    object1.position.x + object1.width >= object2.position.x + 25 &&  
    object1.position.x <= object2.position.x + object2.width - 25 &&  
    object1.position.y + object1.height >= object2.position.y &&  
    object1.position.y <= object2.position.y + object2.height  
  )  
}
```

```
// check victory  
if (toTheDoor({object1: player.hitbox, object2: door}) && this.life != 0) {  
  |   victory()  
}
```

```
//victory  
function victory() {  
  |   endGameMessage.innerText='VICTORY';  
}
```

8 - Victoire

9 - Défaite



```
// check spike collide
let object1 = player.hitbox
for (let i = 0; i < collisionBlocksSpike.length; i++) {
  let object2 = collisionBlocksSpike[i]
  if (
    object1.position.x + object1.width >= object2.position.x &&
    object1.position.x <= object2.position.x + object2.width &&
    object1.position.y + object1.height >= object2.position.y &&
    object1.position.y <= object2.position.y + object2.height
  ) {
    console.log('spike damage')
  }
}

// check if felt into lava
if (player.hitbox.position.y >= 416) {
  this.life = 0
  damage = true
}
```


10 - Conclusion

objectif rempli à moitié

Améliorable

Jouable

Pur JS

