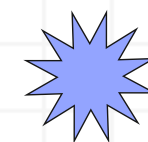




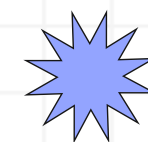
JOGOS EM REACT JS



O QUE É O REACT GAME

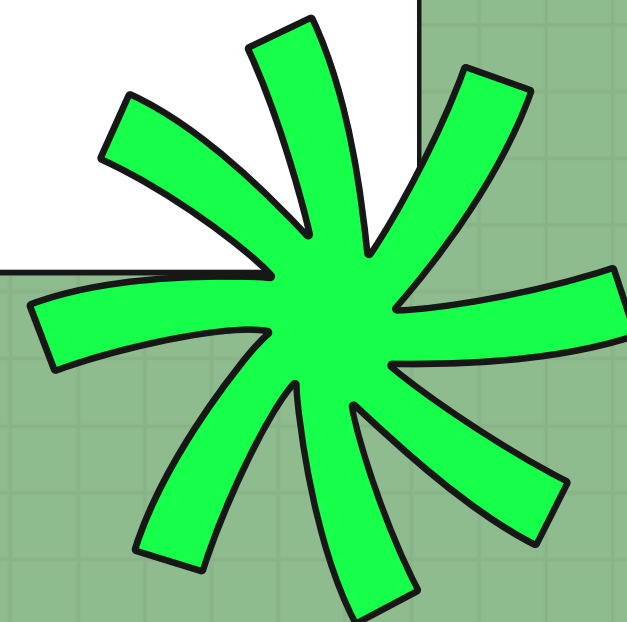
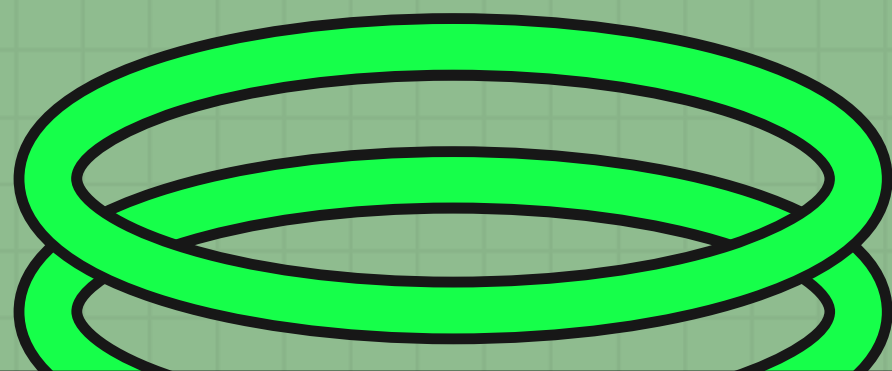


É um game estilo flappy bird que consiste em desviar o avatar SuperIF dos obstáculos.



O jogo também possui regras claras, nesse jogo em específico nós temos as seguintes:

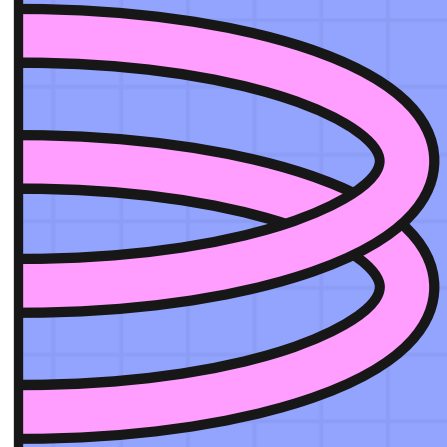
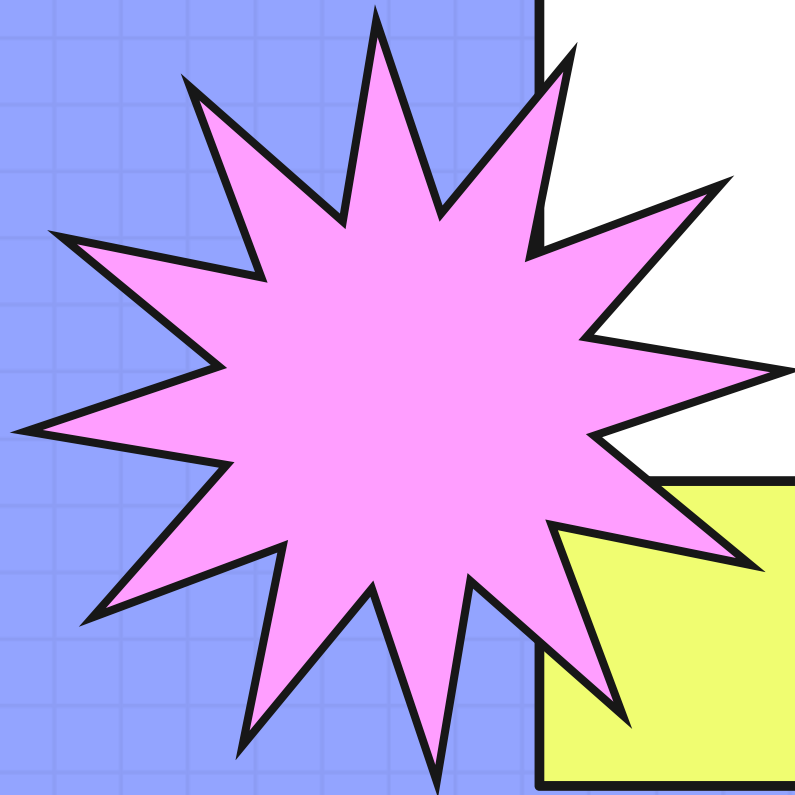
- O jogador está caindo constantemente;
- Há obstáculos indo em direção ao jogador;
- Se o jogador tocar um dos obstáculos ele perde;
- Se o jogador tocar o topo ou o começo da tela ele perde.



Pastas:



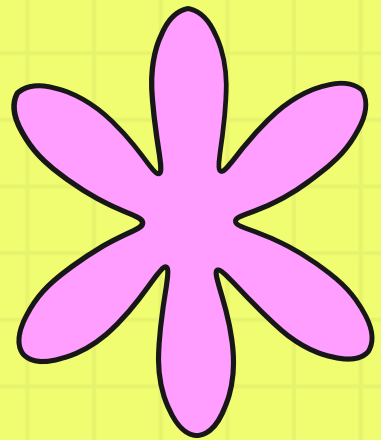
1. index.js
2. background.js
3. player.js
4. Obstacle.js
5. hud.js
6. style.css



```
render() {  
  
  return (  
    <div id="wrapper" ref={(c) => this._wrapper = c}>  
      <BgMusic  
        gameOver={this.state.gameOver}  
        running={this.state.running}  
        mute={this.state.mute}  
      />  
      <Background  
        gameOver={this.state.gameOver}  
        running={this.state.running}  
      >  
        <Hud  
          score={this.state.score}  
          mute={this.state.mute}  
          onMuteToggle={this.onMuteToggle}  
        />  
        <Player  
          running={this.state.running}
```

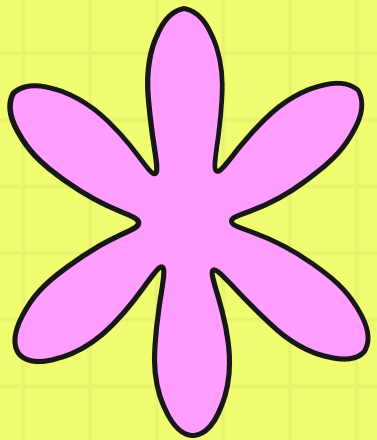
```
        gameOver={this.state.gameOver}  
        style={{  
          left: `${INITIAL_LEFT}px`,  
          top: `${this.state.top}px`  
        }}  
        ref={(c) => { this._player = c }}  
      />  
      {this.state.obstacles.map((o) => (  
        <Obstacle  
          key={o.index}  
          left={o.left}  
          onTop={o.onTop}  
          width={OBSTACLE_WIDTH}  
        />  
      ))}  
    </Background>  
  </div>  
,
```

INDEX.JS – INTERVALO



```
_onInterval () {  
  const now = Date.now()  
  this.setState((oldState) => {  
    let { score, level, lastLeft, obstacleIndex } = oldState  
    const { right, up, down } = this._controls  
    this._resetControls()  
    let obstacleSpeed = INITIAL_OBSTACLE_STEP + INITIAL_OBSTACLE_STEP * (0.2) * level  
    if (right) {  
      obstacleSpeed *= 1 * 6 * Math.min(right, 2)  
    }  
    let top = oldState.top + FALL_STEP  
    if (up) {  
      top -= JUMP_STEP * up  
    }  
    if (down) {  
      top += DROP_STEP * down  
    }  
  
    if (now - this._lastNewObstacles >= NEW_OBSTACLES_TIME) {  
      this._lastNewObstacles = now  
      const numOfObstacles = oldState.obstacles.length  
      if (numOfObstacles) {  
        lastLeft = oldState.obstacles[numOfObstacles - 1].left - INITIAL_OBSTACLE_STEP  
        if (lastLeft <= INITIAL_LEFT + PLAYER_WIDTH + OBSTACLE_MIN_DISTANCE) {  
          lastLeft = INITIAL_LEFT + PLAYER_WIDTH * 3  
        }  
      }  
    }  
  })  
}
```

INDEX.JS – COLISÕES



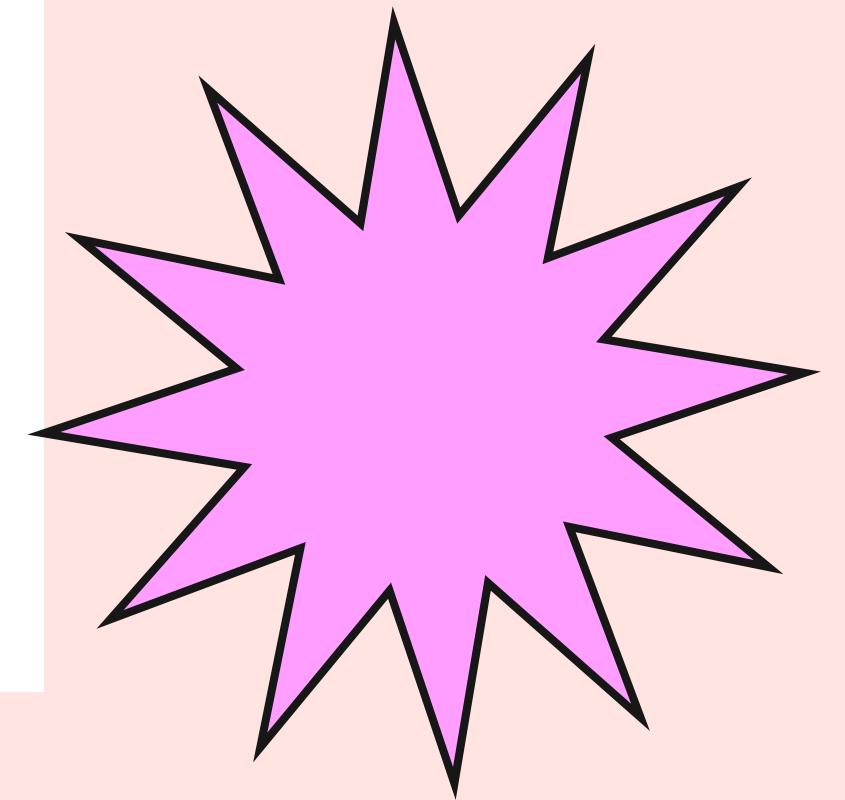
```
_checkColision(playerLeft, playerTop, obstacleData, newLeft) {  
  let isWithingHeight = false  
  const wrapperHeight = this._wrapper.offsetHeight  
  const obstacleHeight = wrapperHeight / 2  
  if (obstacleData.onTop) {  
    if (playerTop <= obstacleHeight) {  
      isWithingHeight = true  
    }  
  } else {  
    if ((playerTop + PLAYER_HEIGHT) >= wrapperHeight - obstacleHeight) {  
      isWithingHeight = true  
    }  
  }  
  if (!isWithingHeight) {  
    return false  
  }  
  
  if (playerLeft >= newLeft &&  
    (playerLeft + PLAYER_WIDTH) <= obstacleData.left + OBSTACLE_WIDTH  
  ) {  
    return true  
  }  
  return false  
}
```


INDEX.JS – "OUVINDO" OS BOTÕES

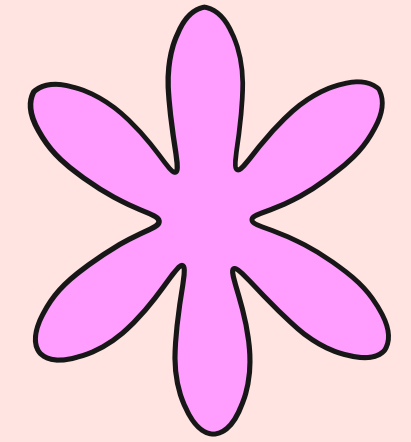
```
1  _listenToKey(ev) {
2    if (ev.code === 'Enter') {
3      if (!this.state.running) {
4        this._startGame()
5      }
6    }
7    if (this.state.running) {
8      if (ev.code === 'Space' || ev.code === 'ArrowUp') {
9        this._controls.up++
10      }
11      if (ev.code === 'AltRight' || ev.code === 'ControlLeft' || ev.code === 'AltLeft' || ev.code === 'ControlRight') {
12        this._controls.down++
13      }
14      if (ev.code === 'ArrowRight') {
15        this._controls.right++
16      }
17    }
18  }
```

BACKGROUND

```
1  import React, { Component } from 'react';
2  import styled from 'styled-components'
3  const Background = styled.div`
4    background-color: blue;
5    width: 100%;
6    height: 100%;
7    overflow: hidden;
8    position: absolute;
9    text-align: center;
10 `
11 const GameOverText = styled.div`
12   color: white;
13   text-align: center;
14   position: absolute;
15   top: 50%;
16   transform: translateY(-50%);
17   width: 100%;
18 `
19 const GameOverBackground = styled(Background)`
20   background-color: red;
```

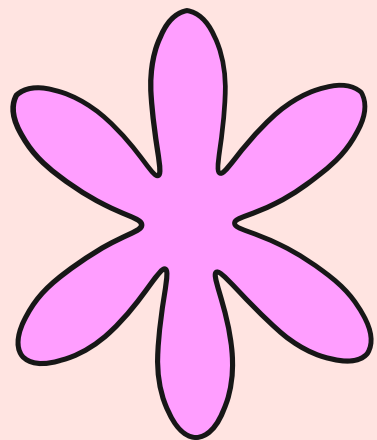


BACKGROUND



```
22  const StartBackground = styled(Background)`
23    background-color: green;
24  `
25  const TopGrass = styled.div`
26    position: absolute;
27    background-image:
28      url("data:image/svg+xml;utf8,<svg xmlns='http://www.w3.org/2000/
29    </svg>"), url("data:image/svg+xml;utf8,<svg xmlns='http://www.w3.org/200
30    </svg>");
31    background-position: left bottom, left top;
32    background-size: 15vh auto;
33    background-repeat: repeat-x;
34    width: 100%;
35    height:100%;
36    z-index: 3;
37
38  `
39  export class Background extends Component {
40    constructor (props) {
```

BACKGROUND



```
41     super(props)
42     this.state = {
43       showGameOver: false
44     }
45   }
46   componentWillReceiveProps (nextProps) {
47     if (nextProps.gameOver && !this.props.gameOver) {
48       this._intervalId = setTimeout(() => {
49         console.log('setting timeout')
50         this.setState({showGameOver: true})
51       }, 2000)
52     }
53     if (nextProps.running && !this.props.running) {
54       clearInterval(this._intervalId)
55       this.setState({showGameOver: false})
56     }
57   }
58   render () {
59     if (this.props.gameOver && this.state.showGameOver) {
60       return (
61         <GameOverBackground {...this.props}>
62           <GameOverText>
63             <h1>Game Over!</h1>
64             <h2>Tente novamente (Enter)</h2>
65           </GameOverText>
66         </GameOverBackground>
67       )

```

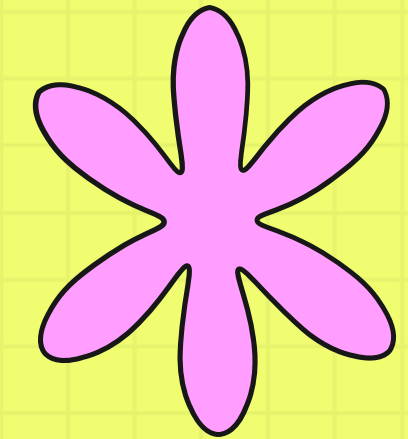
BACKGROUND.JS

```
1  import React, { Component } from 'react';
2  import styled from 'styled-components'
3  const BackGround = styled.div`
4    background-color: blue;
5    width: 100%;
6    height: 100%;
7    overflow: hidden;
8    position: absolute;
9    text-align: center;
10 `
11 constGameOverText = styled.div`
12   color: white;
13   text-align: center;
14   position: absolute;
15   top: 50%;
16   transform: translateY(-50%);
17   width: 100%;
18 `
19 constGameOverBackground = styled(BackGround)`
20   background-color: red;
```

```

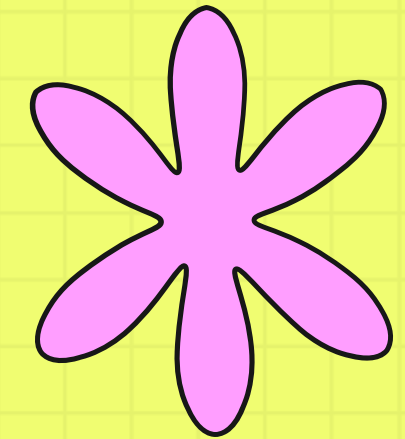
1  import React, { Component } from 'react';
2  import styled, {keyframes} from 'styled-components'
3
4  const gentlyRock = keyframes`
5      0% {transform:rotate(4deg);}
6      50% {transform:rotate(-4deg);}
7      100% {transform:rotate(4deg);}
8  `
9  const dead = keyframes`
10     0% {transform:rotate(0);}
11     50% {transform:rotate(180deg);}
12     100% {transform:rotate(360deg);}
13
14 `
15 const StyledImg = styled.img`
16     width: 100px;
17     height: 100px;
18     position: ${({props}) => (props.running || props.gameOver) ? 'absolute' : 'relative' };
19     left: 0;
20     transition: top 0.2s;
21     z-index: 2;
22     animation: ${({props}) => props.gameOver ? `${dead} linear 0.5s infinite` : `${gentlyRock} lin
23
24 `

```



Player I

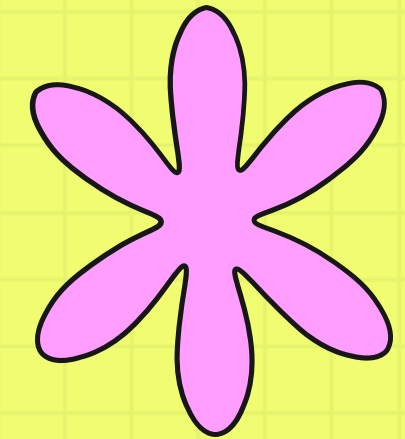
Player II



```
25  export class Player extends Component {  
26    render () {  
27      const svg = ``  
28      return (  
29        <StyledImg  
30          {...this.props}  
31          src="https://openclipart.org/download/75889/duck-line-art-pitr-Ducky-icon.svg" />  
32      )  
33    }  
34  }  
35  export default Player
```

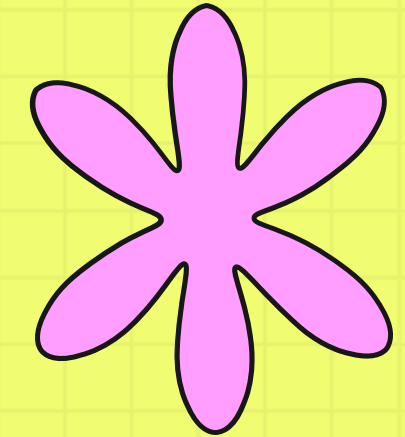
JS obstacle.js ×

```
1  import React, { Component } from 'react';
2  import styled from 'styled-components';
3  const Pipe = styled.div`
4    background-color: green;
5    height: 50%;
6    position: absolute;
7    transition: left 0.2s;
8  `;
9  export class Obstacle extends Component {
10   render() {
11     const style = {
12       left: `${this.props.left}px`,
13       width: `${this.props.width}px`,
14     };
15     if (this.props.onTop) {
16       style.top = '0';
17     } else {
18       style.bottom = '0';
19     }
20     return <Pipe {...this.props} style={style} />;
21   }
22 }
23 export default Obstacle;
24
```



Obstacle

Hud



```
1  import React, { Component } from 'react';
2  import styled from 'styled-components';
3
4  const Container = styled.div`
5    position: fixed;
6    right: 0;
7    top: 0;
8    padding: 1.3rem;
9    font-size: 1.3rem;
10   z-index: 4;
11 `;
12 export class Hud extends Component {
13   render() {
14     return <Container>Score: {this.props.score}</Container>;
15   }
16 }
17 export default Hud;
```

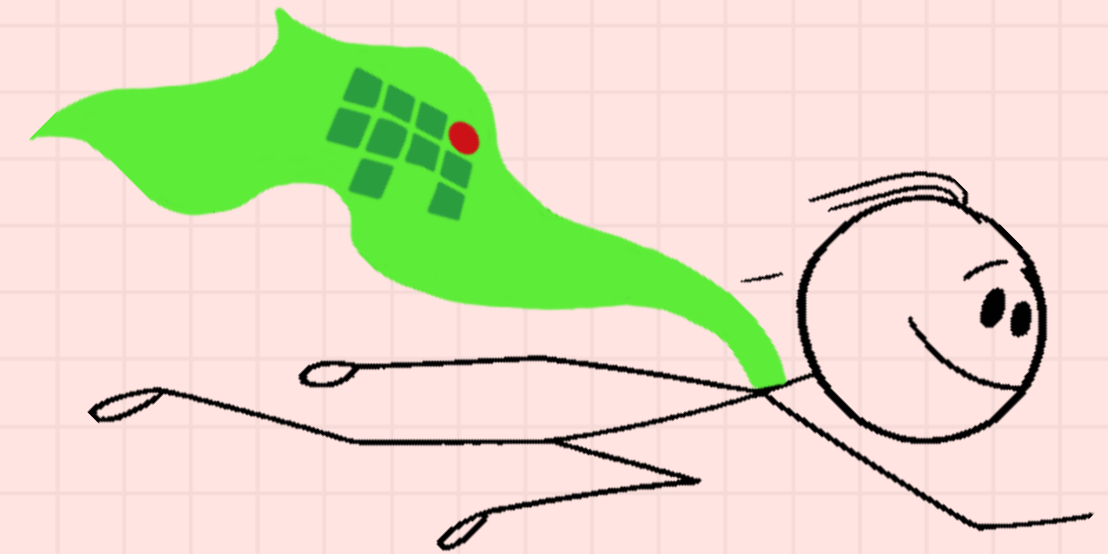
style.css

```
1  √ h1, p {
2    font-family: Lato;
3  }
4  √ html, body, #root, #blitz-app, #root, #wrapper {
5    width: 100%;
6    height: 100%;
7  }
8  √ body {
9    margin: 0;
10 }
11 √ #wrapper {
12   position: relative;
13 }
```



FUNCIONALIDADE

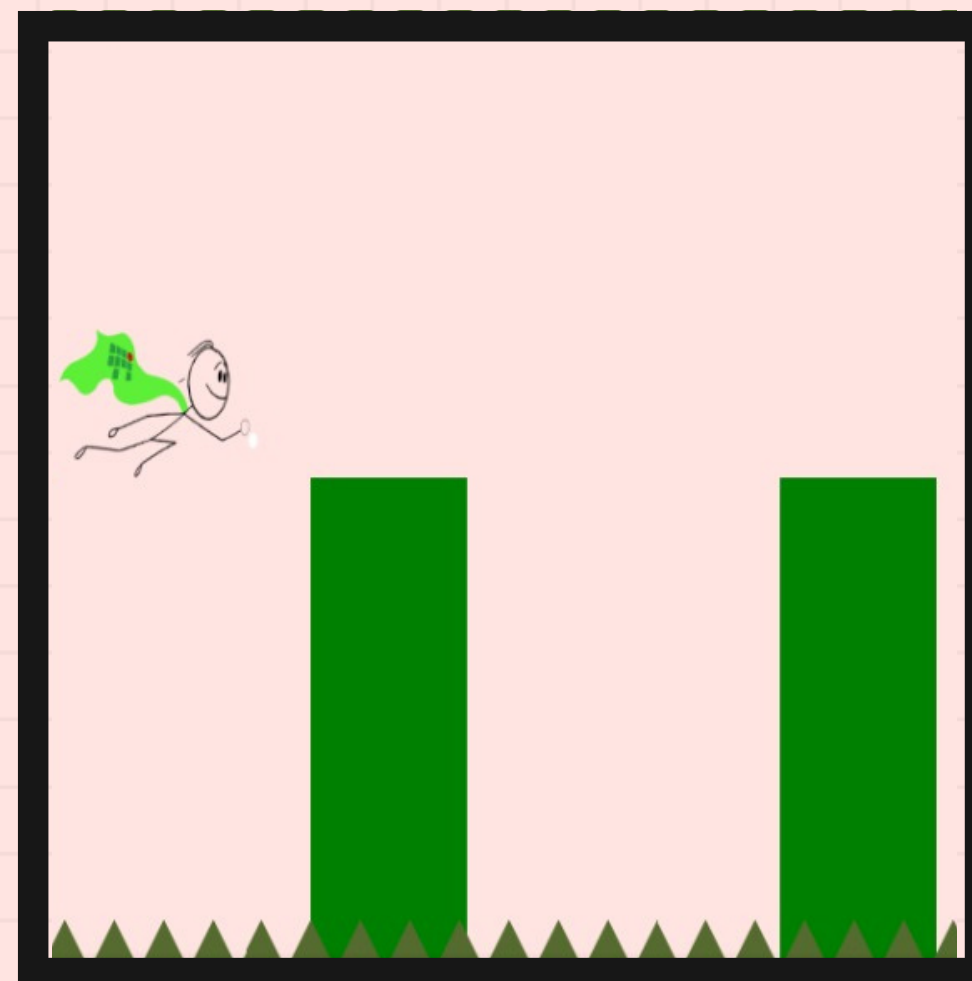
REACT GAME



Início

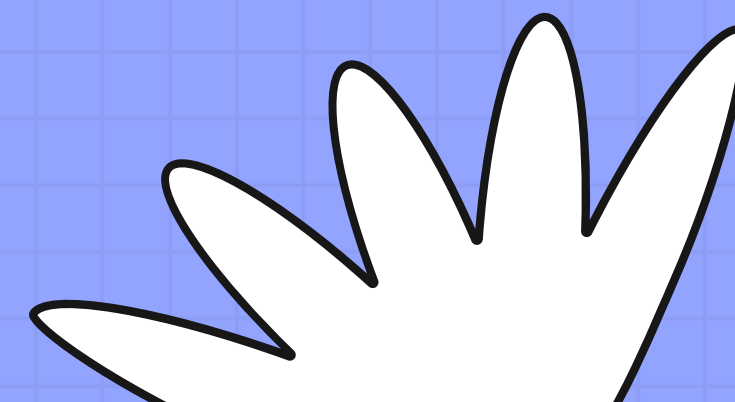
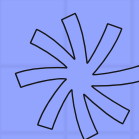
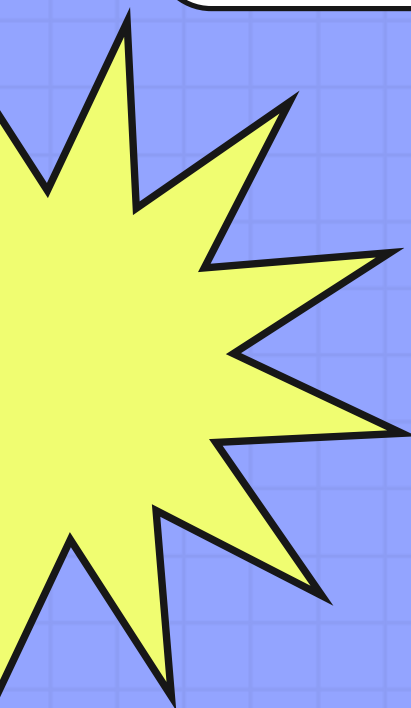
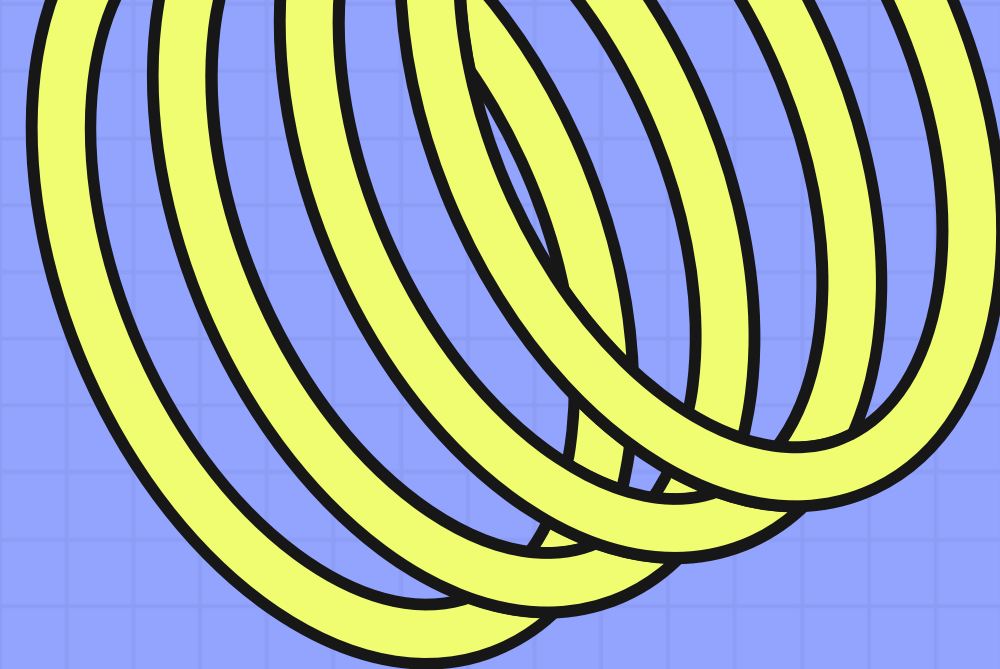
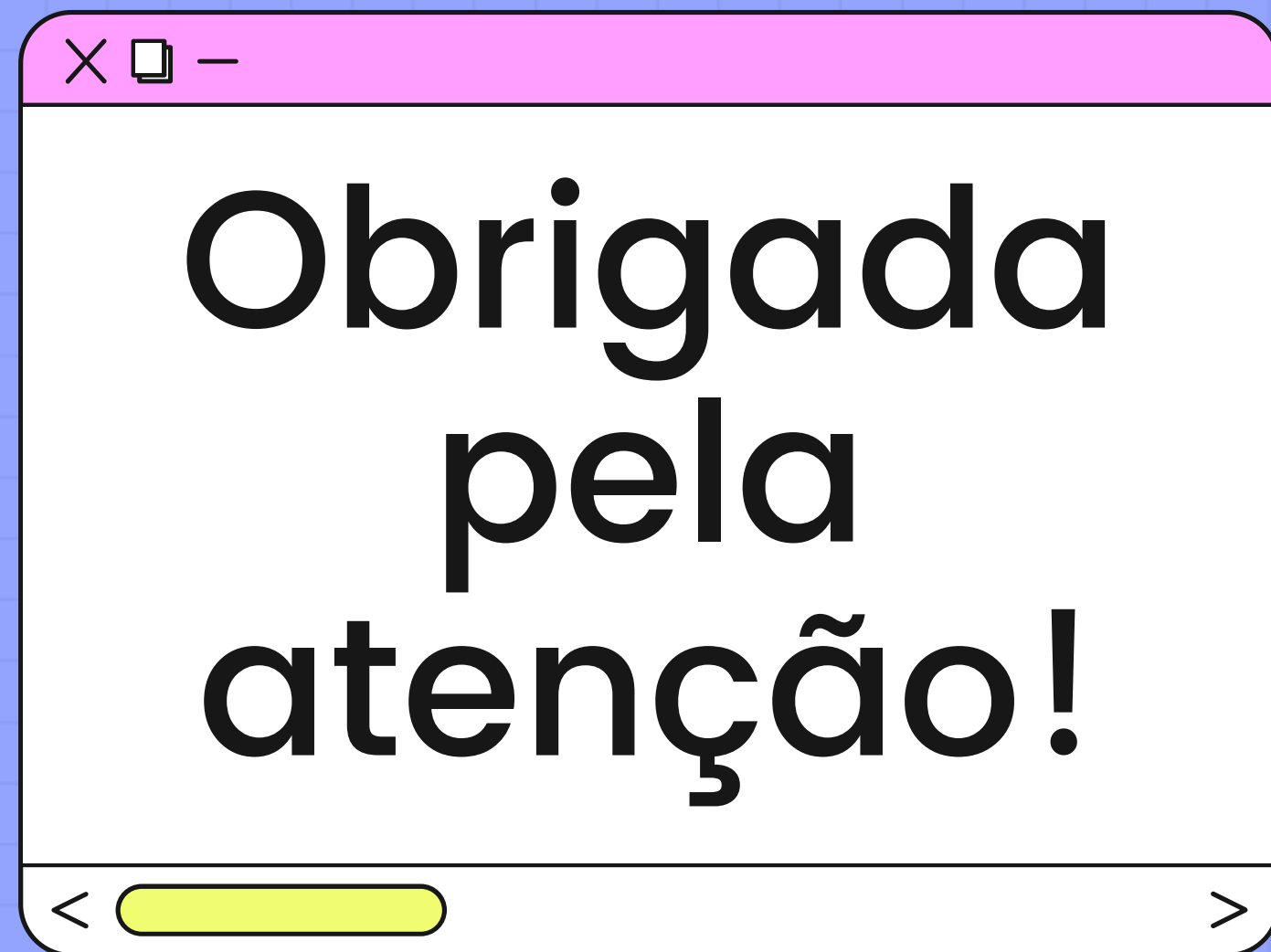


Durante

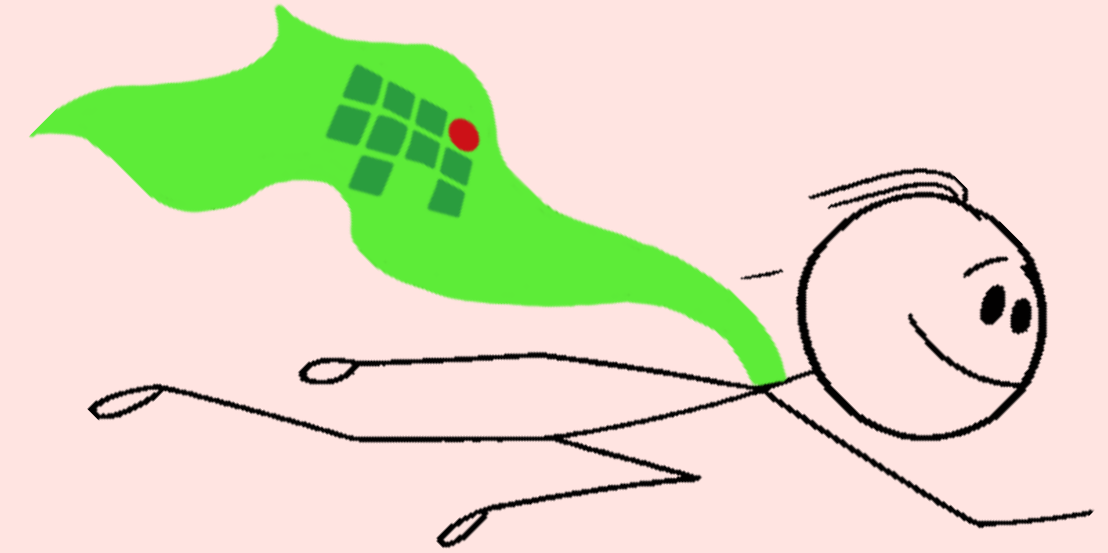


Fim





REFERÊNCIAS:



<https://medium.com/perguntaodev/desenvolvendo-um-jogo-em-react-sem-instalar-nada-38e8964cf3e3>