

```

1  const config = {
2    SCREEN_SIZE: 800,
3    ELEMENT_SIZE: 100,
4    MIDDLE: 800/2 - 100/2,
5    MOVEMENT: 5
6  }
7
8  class Bullet extends React.Component {
9    render() {
10     // console.log('Bullet:render')
11     const style = {
12       width: config.ELEMENT_SIZE,
13       height: config.ELEMENT_SIZE,
14       background: 'pink',
15       position: 'absolute',
16       bottom: this.props.bulletPos,
17       left: this.props.playerPos
18     }
19     return <div style={style} />
20   }
21 }
22
23 class Enemy extends React.Component {
24   constructor(props) {
25     super(props)
26
27     this.state = {
28       display: 'block',
29       left: props.left
30     }
31
32     this.destroyed = false
33   }
34
35   componentWillMount() {
36     let movement = config.MOVEMENT
37
38     const frame = () => {
39       if (this.state.left + config.ELEMENT_SIZE == config.SCREEN_SIZE ||
40 this.state.left === 0) {
41         movement *= -1
42       }
43       this.setState({
44         left: this.state.left + movement
45       })
46
47       const me = {
48         min: {
49           X: this.state.left,
50           Y: this.props.bottom - config.ELEMENT_SIZE
51         },
52         max: {
53           X: this.state.left + config.ELEMENT_SIZE,
54           Y: this.props.bottom
55         }
56       }
57
58       const other = {
59         min: {
60           X: this.props.bulletPos.x,

```

```

60           Y: this.props.bulletPos.y - config.ELEMENT_SIZE
61         },
62         max: {
63           X: this.props.bulletPos.x + config.ELEMENT_SIZE,
64           Y: this.props.bulletPos.y
65         }
66       }
67
68       // colision detection
69       if(!
70         me.max.X < other.min.X ||
71         me.max.Y < other.min.Y ||
72         me.min.X > other.max.X ||
73         me.min.Y > other.max.Y
74       ) {
75         this.handleDestroy()
76       }
77
78       requestAnimationFrame(frame)
79     }
80
81     requestAnimationFrame(frame)
82   }
83
84   handleDestroy() {
85     if (!this.destroyed) {
86       this.destroyed = true
87       console.log('Enemy destroyed')
88
89       this.setState({
90         display: 'none'
91       })
92
93       this.props.onEnemyDestroyed()
94     }
95   }
96
97   render() {
98     // console.log('Enemy:render')
99     const style = {
100       width: config.ELEMENT_SIZE,
101       height: config.ELEMENT_SIZE,
102       background: 'yellow',
103       position: 'absolute',
104       bottom: this.props.bottom,
105       left: this.state.left,
106       display: this.state.display
107     }
108     return <div style={style} />
109   }
110 }
111
112 class Player extends React.Component {
113   render() {
114     // console.log('Player:render')
115     const style = {
116       width: config.ELEMENT_SIZE,
117       height: config.ELEMENT_SIZE,
118       background: 'blue',

```

```

120     position: 'absolute',
121     bottom: 0,
122     left: this.props.position
123   }
124   return <div style={style} />
125 }
126 }
127
128 class Game extends React.Component {
129   constructor() {
130     super()
131
132     this.state = {
133       playerPos: config.MIDDLE,
134       bulletPos: 1100,
135       enemies: 6
136     }
137
138     this.onEnemyDestroyed = this.onEnemyDestroyed.bind(this)
139   }
140
141   shoot() {
142     this.setState({
143       bulletPos: config.ELEMENT_SIZE
144     })
145     const frame = () => {
146       this.setState({
147         bulletPos: this.state.bulletPos + config.MOVEMENT
148       })
149       requestAnimationFrame(frame)
150     }
151     requestAnimationFrame(frame)
152   }
153
154   onEnemyDestroyed() {
155     this.setState({
156       enemies: this.state.enemies - 1
157     })
158     if (this.state.enemies - 1 <= 0) {
159       alert('you win!')
160     }
161   }
162
163   handleKeyDown(e) {
164     if (e.key === "ArrowRight") {
165       this.setState({
166         playerPos: this.state.playerPos + config.MOVEMENT
167       })
168     }
169     if (e.key === "ArrowLeft") {
170       this.setState({
171         playerPos: this.state.playerPos - config.MOVEMENT
172       })
173     }
174     if (e.key === " ") {
175       this.shoot()
176     }
177   }
178
179   render() {

```

```

180     const style = {
181       width: config.SCREEN_SIZE,
182       height: config.SCREEN_SIZE,
183       background: 'red',
184       position: 'relative'
185     }
186
187     const bulletPos = { y: this.state.bulletPos, x: this.state.playerPos }
188
189     return (
190       <div style={style} onKeyDown={this.handleKeyDown.bind(this)}
191       tabIndex="0">
192         <Enemy
193           bulletPos={bulletPos}
194           bottom={config.MIDDLE} left={150}
195           onEnemyDestroyed={this.onEnemyDestroyed}
196         />
197         <Enemy
198           bulletPos={bulletPos}
199           bottom={config.MIDDLE} left={350}
200           onEnemyDestroyed={this.onEnemyDestroyed}
201         />
202         <Enemy
203           bulletPos={bulletPos}
204           bottom={config.MIDDLE} left={550}
205           onEnemyDestroyed={this.onEnemyDestroyed}
206         />
207         <Enemy
208           bulletPos={bulletPos}
209           bottom={config.MIDDLE + 150} left={150}
210           onEnemyDestroyed={this.onEnemyDestroyed}
211         />
212         <Enemy
213           bulletPos={bulletPos}
214           bottom={config.MIDDLE + 150} left={350}
215           onEnemyDestroyed={this.onEnemyDestroyed}
216         />
217         <Enemy
218           bulletPos={bulletPos}
219           bottom={config.MIDDLE + 150} left={550}
220           onEnemyDestroyed={this.onEnemyDestroyed}
221         />
222         <Bullet bulletPos={this.state.bulletPos} playerPos=
223         {this.state.playerPos} />
224         <Player position={this.state.playerPos} />
225       </div>
226     )
227   }
228
229   ReactDOM.render(
230     <Game />,
231     document.getElementById('reactroot')
232   )

```