



# Le front-end

par Anthony Maurial



# Sommaire

- introduction
- les détails techniques
- les avancés

# introduction

## DEVELOPPEUR FRONTEND

Il est chargé de développer  
l'interface visuel d'un site  
ou d'une application web

LesJeudis



# les détails techniques

## Connexion

```
JS autoconnectjs JS scripts x JS autoconnectjs JS scripts
login JS scripts +
1 |import { io } from "https://cdn.socket.io/4.7.5/socket.io.esm.min.js";
2 let loginBtn = document.getElementById('login')
3
4
5 const socket = io("83.195.132.36")
6 const username = document.getElementById("username")
7 const password = document.getElementById("password")
8 const incorrectText = document.getElementById("incorrect")
9
10 loginBtn.addEventListener('click', () => {
11   login()
12 })
13 document.addEventListener("keydown", KE => {
14   if (KE.key === "Enter") {
15     login()
16   }
17 })
18
19 username.addEventListener("input", () => {
20   incorrectText.classList.add("hidden")
21 })
22
23 password.addEventListener("input", () => {
24   incorrectText.classList.add("hidden")
25 })
26
27 function setCookie(cname, cvalue, exdays) {
28   const d = new Date();
29   d.setTime(d.getTime() + (exdays * 24 * 60 * 60 * 1000));
30   let expires = "expires=" + d.toUTCString();
31   document.cookie = cname + "=" + cvalue + ";" + expires + ";path=/";
32 }
33
34 function login() {
35   socket.emit("login.manual", username.value, password.value, (res) => {
36     if (res.success) {
37       setCookie("token", res.cookie, 365)
38       location = "../"
39     } else {
40       console.log("Connexion réussie");
41     }
42     incorrectText.classList.remove("hidden")
43     password.value = ""
44     console.log("Connexion échouée");
45     console.log(res)
46   })
47 }
```

```
JS autoconnectjs JS scripts
1 |import { io } from "https://cdn.socket.io/4.7.5/socket.io.esm.min.js";
2 var socket = io("83.195.132.36")
3 window.socket = socket
4 socket.on("loc.change", (l) => { location = l});
5
6 function getCookie(cname) {
7   let name = cname + "=";
8   let decodedCookie = decodeURIComponent(document.cookie);
9   let ca = decodedCookie.split(';');
10  for (let i = 0; i < ca.length; i++) {
11    let c = ca[i];
12    while (c.charAt(0) == ' ') {
13      c = c.substring(1);
14    }
15    if (c.indexOf(name) == 0) {
16      return c.substring(name.length, c.length);
17    }
18  }
19  return "";
20 }
21
22 if (getCookie("token")) {
23   socket.emit("login.auto", getCookie("token"), (res) => {
24     if (res.success) {
25       location = "../login"
26     } else {
27       console.log(res)
28     }
29   })
30 } else {
31   location = "../login"
32 }
```

# les détails techniques

carte

pillier

user

logs

Caserne de Bordeaux

La mère de lolo est tellement pathétique qu'elle pourrait probablement décrocher un doctorat en médiocrité

FireSense

Caserne de Bordeaux

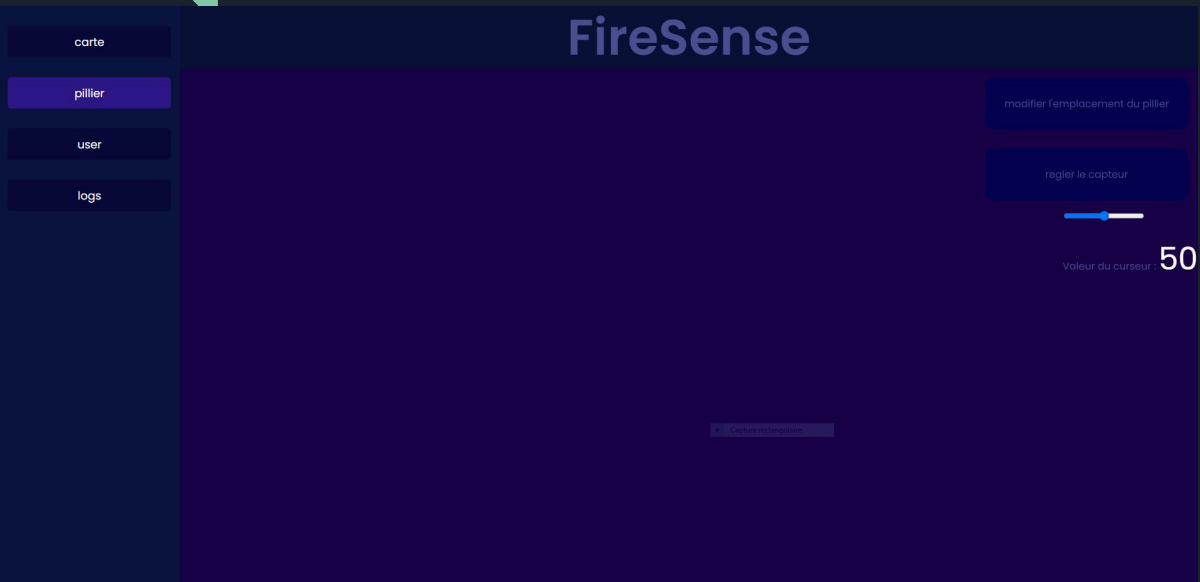
Hors ligne

Proumouvoir admin

Supprimer

```
user > JS script.js > ...
41 }
42
43 function showUser(userId) {
44   let user = users[userId]
45   usernameHTML.innerText = user.name
46   userstatus.innerText = (user.online ? "En ligne" : "Hors ligne")
47   console.log("Cet utilisateur est: ${user.admin ? "Admin" : "Rien"}")
48
49   admbtnHTML.innerHTML = `<span>${user.admin ? "Révoquer admin" : "Proumouvoir admin"}</span>`
50 }
51
52 deletepopupHTML.addEventListener("click", (e) => {
53   if (e.srcElement === deletepopupHTML) {
54     deletepopupHTML.classList.add("hidden")
55   }
56 })
57
58 delbtnHTML.addEventListener("click", () => {
59   if (actual.id) {
60     deletepopupHTML.classList.remove("hidden")
61     popupText.innerText = "Voulez-vous vraiment supprimer ${users[actual.id].name} ?"
62     actualAction = "delete"
63   }
64 })
65 admbtnHTML.addEventListener("click", () => {
66   if (actual.id) {
67     deletepopupHTML.classList.remove("hidden")
68     popupText.innerText = "Voulez-vous vraiment ${users[actual.id].admin ? "révoquer les droits d'admin à" : "proumouvoir"} ${users[actual.id].name} ${users[actual.id].admin} ?"
69     actualAction = "admin.change"
70   }
71 })
72 delCancel.addEventListener("click", () => {
73   deletepopupHTML.classList.add("hidden")
74   actualAction = undefined
75 })
76
77 delConfirm.addEventListener("click", () => {
78   let user = users[actual.id]
79   socket.emit("user.${actualAction}", actual.id, {success} => {
80     console.log(`${actualAction} success: ${success}`);
81     deletepopupHTML.classList.add("hidden")
82   })
83 })
84 }
```

# les détails techniques



```
pillier ▶ JS script.js ▶ socket.emit("pylons.get") callback
19  })
20
21  popupReload.addEventListener("click", () => {
22    location.reload()
23  })
24
25  let bg = new Image()
26  bg.src = "../pillier.png"
27
28  let ptx_selected = {
29    x: 0,
30    y: 0,
31    show: false,
32    ptx: null,
33    mousedown: false,
34  }
35
36  socket.emit("pylons.get", (pt) => {
37    ptx = pt
38    console.log(pt);
39  })
40
41  locationBTN.addEventListener("click", () => {
42    canvas.classList.toggle("hidden")
43  })
44
45  let ptx = {}
46
47  function draw() {
48    requestAnimationFrame(draw)
49    ctx.drawImage(bg, 0, 0)
50
51    if (ptx_selected.show) {
52      drawPoint(ptx_selected.x, ptx_selected.y, 11, "white")
53    }
54    for (const id in ptx) {
55      let pt = ptx[id]
56      drawPoint(pt.pos.x * 10, pt.pos.y * 10, 10)
57    }
58  }
59
60  function drawPoint(x, y, pointSize, color = "red") {
61    ctx.fillStyle = color
62    ctx.beginPath()
63    ctx.arc(x, y, pointSize, 0, Math.PI * 2)
64    ctx.fill()
65  }
66  draw()
67
```

# les détails techniques

## FireSense

carte

pillier

user

logs

[2024-05-05 21:23:55] ex  
[2024-05-05 21:58:23] ex  
[2024-05-05 22:29:18] new TCP client  
[2024-05-05 22:58:28] ex  
[2024-05-05 23:44:09] ex  
[2024-05-06 00:02:06] new TCP client  
[2024-05-06 00:02:07] new TCP client  
[2024-05-06 00:02:18] new TCP client  
[2024-05-06 00:02:23] new TCP client  
[2024-05-06 00:02:28] new TCP client  
[2024-05-06 00:02:33] new TCP client  
[2024-05-06 00:02:39] new TCP client  
[2024-05-06 00:02:44] new TCP client  
[2024-05-06 00:02:51] new TCP client  
[2024-05-06 00:02:56] new TCP client  
[2024-05-06 00:03:01] new TCP client  
[2024-05-06 00:03:06] new TCP client  
[2024-05-06 00:03:11] new TCP client  
[2024-05-06 00:03:16] new TCP client  
[2024-05-06 00:09:56] ex

```
log ▸ JS script.js ▸ ...  
1  const a = document.getElementById("root")  
2  |  
3  socket.emit("logs.get")  
4  socket.on("log.post", (log, i) => {  
5    if (!i) {  
6      return  
7    }  
8    a.innerHTML += `<span class="log">${log}<span>`  
9  })
```

# les avancés

