

my-posts > js > posts.js > onload

```
1 window.onload = function () {
2
3     //Contenido HTML para mostrar error
4     var textoError = `
5         <div class="header">
6             <h2>Error</h2>
7             <i class="fa-solid fa-circle-xmark" onclick="cierraError()"></i>
8         </div>
9         <hr>
10        <h3>No se ha podido realizar la consulta debido a un error:</h3>
11        <p class="message1">Póngase en contacto con el área de soporte IT indicando el
12        siguiente mensaje:</p>
13    `;
14
15    document.querySelector("header i").addEventListener("click", leeUsuarios);
16
17    //Manejador de evento ficha usuarios colocado en body. Se resuelve por traversing
18    const sectionUsuarios = document.querySelector("section");
19    sectionUsuarios.addEventListener("click", accionEnFicha);
20
21    async function leeUsuarios() {
22        // fetch("https://jsonplaceholder.typicode.com/userskk")
```

my-posts > js > posts.js > onload

```
17 //Manejador de evento ficha usuarios colocado en body. Se resuelve por traversing
18 const sectionUsuarios = document.querySelector("section");
19 sectionUsuarios.addEventListener("click", accionEnFicha);
20
21 async function leeUsuarios() {
22     // fetch("https://jsonplaceholder.typicode.com/userskk")
23     // .then(response => response.json())
24     // .then(users => {
25     //     procesaUsuarios(users)
26     // })
27     // .catch((error) => document.write(error))
28
29     try {
30         const response = await fetch("https://jsonplaceholder.typicode.com/users");
31         const usuarios = await response.json();
32         procesaUsuarios(usuarios);
33     } catch (error) {
34         //Mensaje estandar de error
35         let errorDiv = document.createElement("div");
36         errorDiv.classList.add("error");
37         //Obtenemos message de error
38         let spanMesgError = document.createElement("span");
39         spanMesgError.innerHTML = `<span>ERROR: ${error.message}</span>`;
```

```
my-posts.html JS posts.js X
my-posts > js > JS posts.js > onload
23 // .then(response => response.json())
24 // .then(users => {
25 //     procesaUsuarios(users)
26 // })
27 // .catch((error) => document.write(error))
28
29 try { const response: Response
30     const response = await fetch("https://jsonplaceholder.typicode.com/users");
31     const usuarios = await response.json();
32     procesaUsuarios(usuarios);
33 } catch (error) {
34     //Mensaje estandar de error
35     let errorDiv = document.createElement("div");
36     errorDiv.classList.add("error");
37     //Obtenemos message de error
38     let spanMsgError = document.createElement("span");
39     spanMsgError.innerHTML = `<span>ERROR: ${error.message}</span>`;
40     errorDiv.innerHTML = textoError;
41     errorDiv.appendChild(spanMsgError);
42     sectionUsuarios.appendChild(errorDiv);
43 }
44
45 function procesaUsuarios(usuarios) {
46     //Colocamos los usuarios en sus fichas. El id lo colocamos como atributo
```

```
my-posts.html JS posts.js X
my-posts > js > JS posts.js > onload > leeUsuarios
43 }
44
45 function procesaUsuarios(usuarios) {
46     //Colocamos los usuarios en sus fichas. El id lo colocamos como atributo
47     data-id para su lectura en la selección para mostrar posts. Limpiamos el div
48     para no repetir posts
49     sectionUsuarios.innerHTML = "";
50     for (let usuario of usuarios) {
51         let cadenaHTML = `
52         <div class="usuario">
53             <h4 data-id="${usuario.id}">${usuario.username.substring(0, 2)}</h4>
54             <p class="username">${usuario.username}</p>
55             <p><strong>Nombre:</strong> ${usuario.name}</p>
56             <p><strong>Email:</strong> ${usuario.email}</p>
57             <p><strong>Dirección:</strong> ${usuario.address.street} - ${usuario.
58             address.city}</p>
59             <p><strong>Compañía:</strong> ${usuario.company.name}</p>
60             <div class="posts"></div>
61         </div>
62         `;
63         sectionUsuarios.innerHTML += cadenaHTML;
64     }
65 }
```

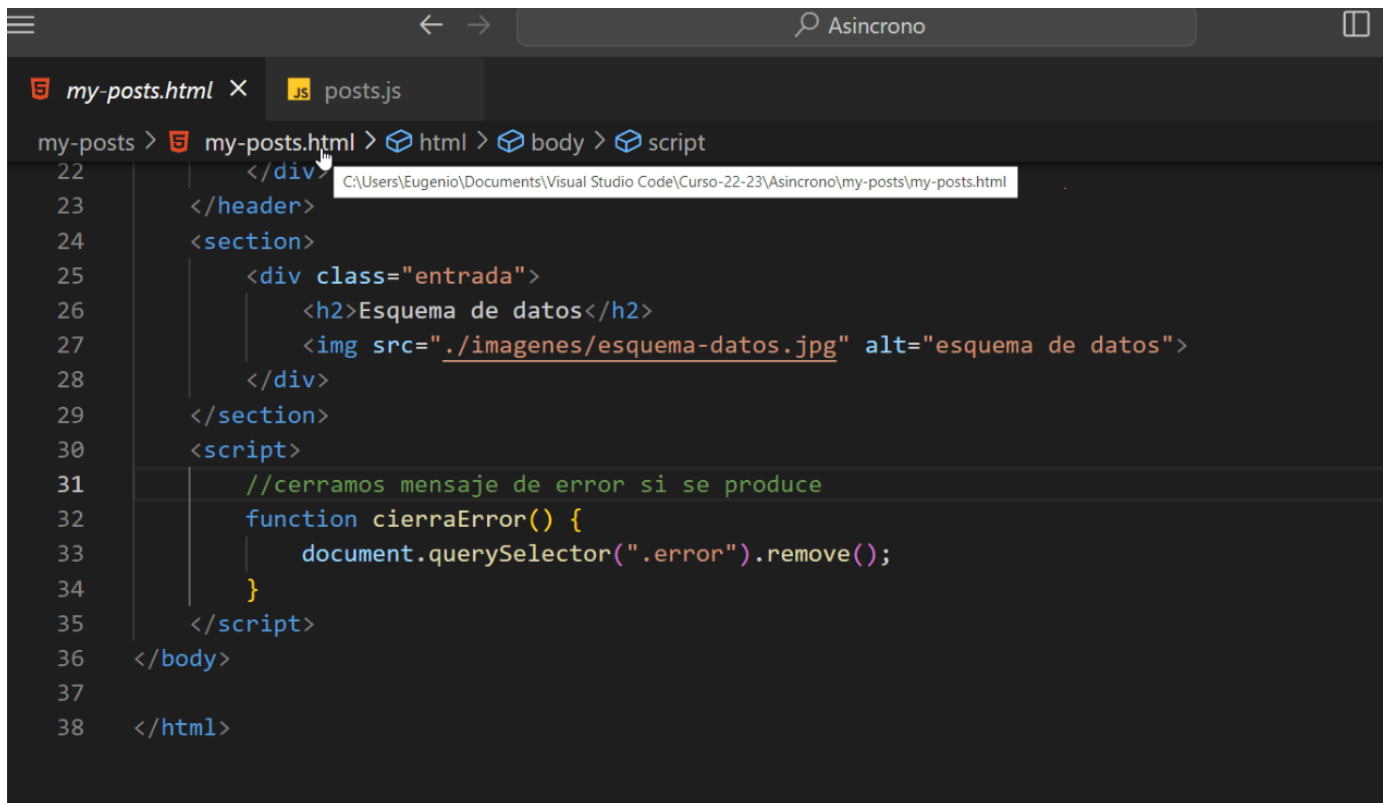
```
my-posts.html JS posts.js X
my-posts > js > JS posts.js > onload > accionEnFicha
62     }
63 }
64 //Función que controla en qué parte de la ficha se hace click:
    usuario o post. Se toma la acción correspondiente. Se realiza
    traversing
65 function accionEnFicha(event) {
66     console.log("Acción en Ficha...");
67
68     //Selección de usuario para mostrar posts
69     //Comprobamos que clicamos en el H4 de usuario y accedemos al
    atributo data-id
70     if (event.target.nodeName == "H4") {
71         let usuarioId = event.target.getAttribute("data-id");
72         //Guardamos referencia del div posts
73         let divPosts = event.target.parentNode.children[6];
74         seleccionUsuario(usuarioId, divPosts)
75         return;
76     }
77
78     //Selección de post para mostrar comments
79     if (event.target.classList.contains("titulo-post")) {
80         let postId = event.target.previousElementSibling.
            innerText;
81         console.log("Post: " + postId)
```

```
my-posts.html JS posts.js X
my-posts > js > JS posts.js > onload > seleccionUsuario > url
82     seleccionPost(postId)
83 }
84 }
85
86 async function seleccionUsuario(usuarioId, divPosts) {
87     //Leemos Posts del usuario seleccionado
88     let url = `https://jsonplaceholder.typicode.com/posts?userId=${usuarioId}`
89     const response = await fetch(url);
90     const posts = await response.json();
91     //Llamamos procesaPosts pasando los posts y la referencia al div posts
92     procesaPosts(posts, divPosts);
93 }
94
95 function procesaPosts(posts, divPosts) {
96     //Si los posts están ya puestos los quitamos
97     //Comprobamos en ancho de divPosts para restaurar valores sin posts
98     if (divPosts.parentNode.style.width == "90%") {
99         divPosts.parentNode.style.width = "420px";
100         divPosts.innerHTML = "";
101         return;
102     }
103     //Si no están puestos los posts, los ponemos y cambiamos ancho del div
104     for (let post of posts) {
```

```
my-posts > js > posts.js > onload > accionEnFicha > divPosts
96 //Si los posts están ya puestos los quitamos
97 //Comprobamos en ancho de divPosts para restaurar valores sin posts
98 if (divPosts.parentNode.style.width == "90%") {
99     divPosts.parentNode.style.width = "420px";
100     divPosts.innerHTML = "";
101     return;
102 }
103 //Si no están puestos los posts, los ponemos y cambiamos ancho del div
104 for (let post of posts) {
105     let cadenaHTML = `
106         <div class="post">
107             <div class="texto-post">
108                 <span>${post.id}</span>
109                 <span class="titulo-post">${post.title}</span>
110             </div>
111             <p>${post.body}</p>
112         </div>
113     `;
114     divPosts.innerHTML += cadenaHTML;
115     divPosts.parentNode.style.width = "90%";
116     divPosts.parentNode.children[0].style.left = "calc(50% - 25px)";
117 }
118 }
```

```
my-posts > js > posts.js > onload > procesaPosts
106     <div class="post">
107         <div class="texto-post">
108             <span>${post.id}</span>
109             <span class="titulo-post">${post.title}</span>
110         </div>
111         <p>${post.body}</p>
112     </div>
113     `;
114     divPosts.innerHTML += cadenaHTML;
115     divPosts.parentNode.style.width = "90%";
116     divPosts.parentNode.children[0].style.left = "calc(50% - 25px)";
117 }
118 }
119
120 async function seleccionPost(postId) {
121     //Leemos Commnets del Post seleccionado
122     let url = `https://jsonplaceholder.typicode.com/posts/${postId}/comments`
123     const response = await fetch(url);
124     const comments = await response.json();
125     //Llamamos procesaPosts pasando los posts y la referencia al div posts
126     // procesaPosts(posts, divPosts);
127     console.log(comments);
128 }
```

html



The screenshot shows a code editor with two tabs: `my-posts.html` and `posts.js`. The `my-posts.html` tab is active, and the breadcrumb navigation shows the path: `my-posts > my-posts.html > html > body > script`. The code is as follows:

```
22     </div>
23 </header>
24 <section>
25     <div class="entrada">
26         <h2>Esquema de datos</h2>
27         
28     </div>
29 </section>
30 <script>
31     //cerramos mensaje de error si se produce
32     function cierraError() {
33         document.querySelector(".error").remove();
34     }
35 </script>
36 </body>
37
38 </html>
```