EJEMPLOS JS – FUNCIONES BÁSICAS.

---

1. Aceptar cookies y comunicarse con el servidor

```
// Función para aceptar cookies
function acceptCookies() {
  // Usamos fetch para enviar una petición al servidor y decir que aceptamos las cookies
  fetch("/accept-cookies", {
    method: "POST", // Usamos el método POST porque estamos enviando datos
    headers: {
      "Content-Type": "application/json", // Indicamos que enviamos datos en formato
JSON
    },
    body: JSON.stringify({ accepted: true }), // Enviamos un objeto con la propiedad
accepted en true
  })
  .then(response => {
    if (response.ok) {
      // Si todo va bien, ocultamos la barra de cookies
      document.getElementById("cookie-banner").style.display = "none";
    } else {
      // Si hay un error, lo mostramos en la consola
      console.error("Error al aceptar cookies");
    }
  })
  .catch(error => {
    // Capturamos cualquier error de red y lo mostramos
    console.error("Error de red:", error);
  });
}
```

---

2. Rechazar cookies con redirección

```
// Función para rechazar cookies
function rejectCookies() {
  // Redirigimos al usuario a Google si rechaza las cookies
  window.location.href = "https://www.google.es";
}
```

---

3. Comprobar si hay que mostrar la barra de cookies

```
// Función para saber si tenemos que mostrar la barra de cookies
function checkCookies() {
  // Hacemos una petición al servidor para preguntar si debemos mostrar la barra
  fetch("/check-cookies", { method: "GET" })
    .then(response => response.json()) // Convertimos la respuesta a JSON
    .then(data => {
      if (data.showBanner) {
        // Si el servidor dice que debemos mostrar la barra, la hacemos visible
        document.getElementById("cookie-banner").style.display = "block";
      } else {
        // Si no, ocultamos la barra
        document.getElementById("cookie-banner").style.display = "none";
      }
    })
    .catch(error => {
      // Si hay un error, lo mostramos en la consola
      console.error("Error comprobando cookies:", error);
    });
}
```

---

4. Crear la barra de cookies dinámicamente -> ESTE MEJOR NO USARLO

```
// Función para crear la barra de cookies en la página
function createCookieBanner() {
  // Creamos un nuevo elemento <div> para la barra de cookies
  const banner = document.createElement("div");
  banner.id = "cookie-banner"; // Le damos un id para poder manipularlo después

  // Agregamos estilo básico con CSS
  banner.style.position = "fixed";
  banner.style.bottom = "0";
  banner.style.width = "100%";
  banner.style.backgroundColor = "#333";
  banner.style.color = "#fff";
  banner.style.textAlign = "center";
  banner.style.padding = "1rem";
  banner.style.zIndex = "1000";
  banner.style.display = "none"; // Ocultamos la barra inicialmente

  // Agregamos el contenido: texto y botones
  banner.innerHTML = `
    <p>Usamos cookies para mejorar tu experiencia.
      <button onclick="acceptCookies()">Aceptar</button>
      <button onclick="rejectCookies()">Rechazar</button>
    </p>
  `;
```

```
  // Añadimos la barra al final del <body> de la página
  document.body.appendChild(banner);
}
```

---

5. Validar login y enviar datos al servidor

```
// Función para validar los datos del login y enviarlos al servidor
function validateAndSubmitLogin(event) {
  event.preventDefault(); // Evitamos que el formulario recargue la página

  // Obtenemos los valores del formulario
  const username = document.getElementById("username").value.trim();
  const password = document.getElementById("password").value.trim();

  // Comprobamos que los campos no están vacíos
  if (!username || !password) {
    alert("Por favor, rellena todos los campos.");
    return; // Salimos de la función si faltan datos
  }

  // Enviamos los datos al servidor
  fetch("/login", {
    method: "POST", // Usamos POST para enviar datos
    headers: {
      "Content-Type": "application/json", // Indicamos que enviamos JSON
    },
    body: JSON.stringify({ username, password }), // Convertimos los datos a JSON
  })
  .then(response => {
    if (response.ok) {
      return response.json(); // Si todo va bien, obtenemos los datos de la respuesta
    } else {
      throw new Error("Login fallido"); // Si hay un error, lanzamos una excepción
    }
  })
  .then(data => {
    // Redirigimos al usuario a la URL que nos devuelve el servidor
    window.location.href = data.redirectUrl || "/";
  })
  .catch(error => {
    // Si hay un error, mostramos un mensaje al usuario
    alert("Error: " + error.message);
  });
}
```

---

6. Cambiar el nombre de la tienda desde el servidor

```
// Función para cargar el nombre de la tienda desde el servidor
function loadStoreName() {
  // Hacemos una petición al servidor para obtener el nombre de la tienda
  fetch("/store-name", { method: "GET" })
    .then(response => response.json()) // Convertimos la respuesta a JSON
    .then(data => {
      if (data.storeName) {
        // Cambiamos el título de la página y el nombre de la tienda
        document.title = `${data.storeName} - Tienda`;
        document.getElementById("store-name").textContent = data.storeName;
      }
    })
    .catch(error => {
      // Si hay un error, lo mostramos en la consola
      console.error("Error cargando el nombre de la tienda:", error);
    });
}
```

- Las funciones usan **fetch** para comunicarse con el backend.

¿CÓMO AÑADIMOS – UNIMOS JS CON HTML?

Usar el atributo id es suficiente para conectar el HTML con el JavaScript. En JavaScript, puedes acceder a los elementos HTML utilizando document.getElementById("id_del_elemento"), y luego manipularlos o añadir eventos.

Aquí tienes un ejemplo sencillo de cómo el id conecta el HTML con las funciones en JavaScript:

HTML

```
<!DOCTYPE html>
<html lang="es">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Ejemplo</title>
</head>
<body>
  <!-- Barra de cookies -->
  <div id="cookie-banner">
    <p>Usamos cookies para mejorar tu experiencia.
      <button id="accept-btn">Aceptar</button>
      <button id="reject-btn">Rechazar</button>
    </p>
  </div>

  <!-- Nombre de la tienda -->
  <h1 id="store-name">Nombre por defecto</h1>

  <!-- Login -->
  <form id="login-form">
    <input id="username" type="text" placeholder="Usuario">
    <input id="password" type="password" placeholder="Contraseña">
    <button type="submit">Iniciar sesión</button>
  </form>

  <!-- Script externo -->
  <script src="script.js"></script>
</body>
</html>
```

JavaScript (script.js)

```
// Acceder y manipular el banner de cookies
document.getElementById("accept-btn").addEventListener("click", acceptCookies);
document.getElementById("reject-btn").addEventListener("click", rejectCookies);
// Acceder al formulario de login y añadir un evento
document.getElementById("login-form").addEventListener("submit",
validateAndSubmitLogin);

// Cambiar el nombre de la tienda dinámicamente
window.onload = loadStoreName; // Llamamos a la función cuando cargue la página
```

---

1. **Acceso mediante id:**
    - Por ejemplo, document.getElementById("cookie-banner") accede al div que contiene el banner de cookies.
    - document.getElementById("accept-btn") accede al botón de "Aceptar" en el banner.
2. **Añadir eventos:**
    - Usamos addEventListener para vincular funciones JavaScript (como acceptCookies o rejectCookies) a eventos como el clic en los botones.
3. **Manipular contenido:**
    - Para cambiar el contenido del h1 con el id store-name, puedes usar document.getElementById("store-name").textContent = "Nuevo nombre de la tienda";.
4. **Enlace con el backend:**
    - Los datos del HTML se procesan en las funciones JavaScript, y estas interactúan con el servidor mediante peticiones como fetch.