**ÍNDICE**

1. **Introducción**

* Propósito del informe.

1. **FASE 1: CALCULADORA** 
   * Introducción
     + Descripción del artículo sobre la construcción de la calculadora.
     + Requisitos previos para el proyecto.
   * Construcción de la calculadora
     + Estructura de la calculadora.
     + Uso de CSS Grid para diseño y responsividad.
   * Funcionalidad de la calculadora
     + Operaciones básicas y complejas.
   * Conclusión
     + Beneficios y aprendizajes del proyecto de la calculadora.
2. **FASE 2: API DE PROVINCIAS**

* Explicación del código de las provincias
  + Descripción del código.
  + Tecnologías utilizadas.
  + Funcionamiento del código.
* Código
  + Detalle de la solicitud a la API y el manejo de datos

.

1. **Conclusiones**

* Aprendizajes y aplicaciones prácticas de ambos códigos.

**Introducción**

En este informe, se analizan dos códigos: una calculadora implementada en HTML, CSS y JavaScript, y una aplicación que utiliza una API para mostrar información sobre las provincias de España. Se examinarán las partes de cada código, las tecnologías utilizadas y su funcionamiento.

**FASE 1: CALCULADORA**

**Introducción**

El artículo, escrito por Rafael D. Hernandez y publicado el 22 de diciembre de 2022 en FreeCodeCamp, es una guía completa para construir una calculadora HTML desde cero utilizando JavaScript. Este proyecto es una excelente oportunidad para los desarrolladores que buscan mejorar sus habilidades en JavaScript y aprender cómo se pueden utilizar estos conceptos para crear una aplicación funcional.

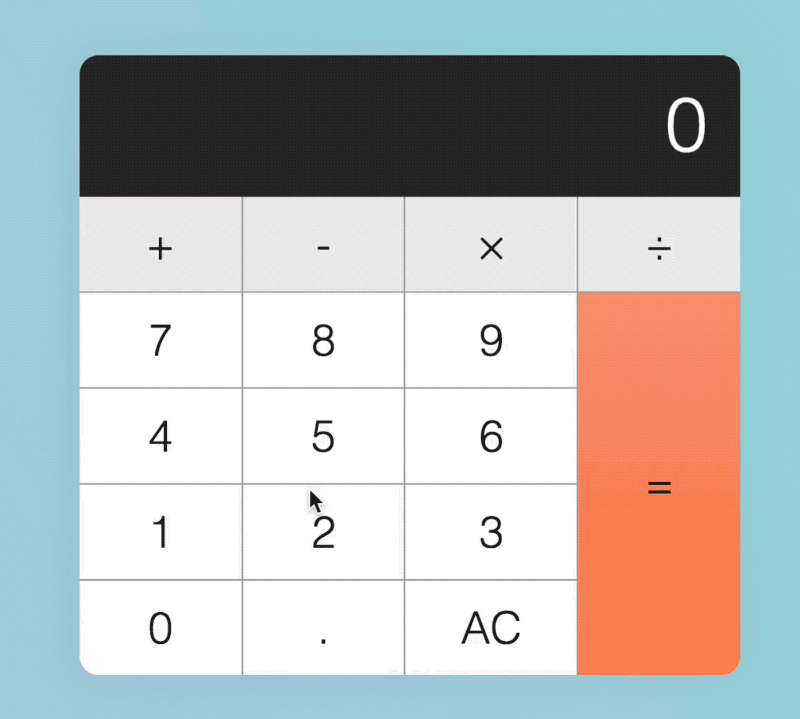
**Requisitos previos**

Antes de comenzar con el proyecto, es esencial tener un conocimiento sólido de JavaScript. Debes estar familiarizado con las sentencias if/else, los bucles for, las funciones de JavaScript, las funciones de flecha, los operadores && y ||, cómo cambiar el texto con la propiedad textContent y cómo agregar oyentes de eventos con el patrón de delegación de eventos. Estos son los bloques de construcción fundamentales de cualquier aplicación JavaScript y son esenciales para la construcción de la calculadora.

**Construcción de la calculadora**

La calculadora consta de dos partes principales: la pantalla y las teclas. La pantalla es donde se muestran los números y los resultados de las operaciones. Las teclas son los botones que el usuario puede presionar para realizar operaciones. Cada tecla tiene un atributo de acción de datos que describe su función. Por ejemplo, las teclas numéricas tienen un atributo de acción de datos de ‘number’, las teclas de operador tienen un atributo de acción de datos de ‘operator’, y así sucesivamente.

Se utiliza CSS Grid para organizar las teclas en un formato de cuadrícula. Esto permite una disposición flexible y fácil de las teclas, lo que facilita la adición o eliminación de teclas en el futuro. Además, el uso de CSS Grid permite que la calculadora sea responsiva, lo que significa que se ajustará automáticamente al tamaño de la pantalla del dispositivo.



**Funcionalidad de la calculadora**

La calculadora puede realizar varias operaciones. Cuando el usuario presiona una tecla numérica, el número correspondiente se muestra en la pantalla. Cuando el usuario presiona una tecla de operador, la calculadora realiza la operación correspondiente.

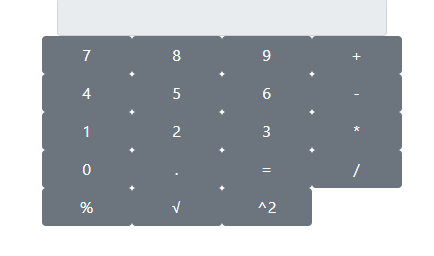
Por ejemplo, si el usuario presiona la tecla ‘+’, la calculadora suma los números. Si el usuario presiona la tecla ‘-’, la calculadora resta los números. Del mismo modo, si el usuario presiona la tecla ‘x’, la calculadora multiplica los números, y si el usuario presiona la tecla ‘÷’, la calculadora divide los números.

Además de las operaciones básicas, la calculadora también puede manejar operaciones más complejas, como la realización de operaciones en cadena. Esto significa que si el usuario introduce una serie de operaciones, como ‘5 + 5 - 2’, la calculadora realizará las operaciones en el orden correcto y mostrará el resultado correcto.

**Conclusión**

El proyecto de la calculadora HTML con JavaScript es una excelente manera de aprender y practicar JavaScript. Te permite entender cómo funcionan las calculadoras y cómo puedes implementar una usando JavaScript. Aunque el proyecto puede parecer desafiante al principio, con paciencia y práctica, podrás construir tu propia calculadora desde cero.

La calculadora nueva, he cambiado el CSS para utilizar boostrap.



Se vería tal que así

**EXPLICACIÓN DEL CÓDIGO DE LA CALCULADORA**

**Descripción del Código**

El código de la calculadora consta de un archivo HTML que define la estructura de la calculadora y un archivo JavaScript que implementa la lógica para realizar cálculos básicos. Se utiliza la biblioteca Bootstrap para el diseño.

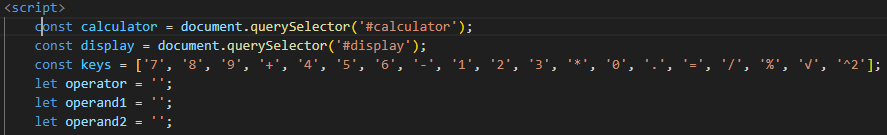
**Tecnologías Utilizadas**

1. HTML
2. CSS (Bootstrap)
3. JavaScript

**Funcionamiento**

La calculadora muestra un display y botones para números y operadores. Al hacer clic en los botones, se van agregando los valores al display. Cuando se presiona el botón "=" se realiza el cálculo según el operador seleccionado y se muestra el resultado en el display.

**Código**



**const calculator = document.querySelector('#calculator');** selecciona el elemento con el ID "calculator" en el HTML y lo almacena en la variable calculator.

**const display = document.querySelector('#display'**); selecciona el elemento con el ID "display" en el HTML y lo almacena en la variable display.

**const keys = [...]** define una matriz con todos los botones que queremos tener en nuestra calculadora.

**let operator = '';, let operand1 = '';, y let operand2 = ''**; son variables que usaremos más adelante para realizar cálculos y rastrear qué números y operadores ha seleccionado el usuario



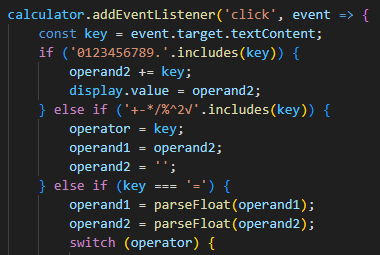
**keys.forEach(...)** itera sobre cada elemento de la matriz keys.

**document.createElement('button')** crea un nuevo elemento de botón en el DOM.

**button.textContent = key**; establece el contenido de texto del botón con el valor actual de key.

**button.className = 'btn btn-secondary col-3'**; agrega las clases de Bootstrap para dar estilo al botón.

**calculator.appendChild(button)**; añade el botón creado al contenedor calculator.



**calculator.addEventListener('click',** ...) añade un escuchador de eventos al contenedor calculator, que detecta los clics dentro de la calculadora.

**const key = event.target.textContent**; obtiene el texto del botón que ha sido clicado.

El bloque **if-else** verifica qué botón ha sido clicado y toma acciones correspondientes:

* Si es un número o un punto decimal, se añade al segundo operando (operand2) y se muestra en el campo de entrada (display).
* Si es un operador matemático, se guarda como operador y se mueve el valor de operand2 a operand1.
* Si el usuario hace clic en el signo igual (=), se realiza la operación matemática correspondiente y se muestra el resultado en la pantalla.

**FASE 2: API de provincias**

**EXPLICACIÓN DEL CÓDIGO DE LAS PROVINCIAS**

**Descripción del Código**

El código utiliza JavaScript para hacer una solicitud a una API que devuelve información sobre las provincias de España. Esta información se muestra en una página HTML, donde cada provincia es un elemento interactivo que muestra más detalles al hacer clic.

**Tecnologías Utilizadas**

1. HTML
2. CSS
3. JavaScript
4. API Fetch

**Funcionamiento**

El código hace una solicitud a la API de El Tiempo que devuelve datos sobre las provincias de España. Luego, crea elementos HTML dinámicamente para mostrar la información de cada provincia. Al hacer clic en una provincia, se muestra más detalle sobre la misma.

**Código**

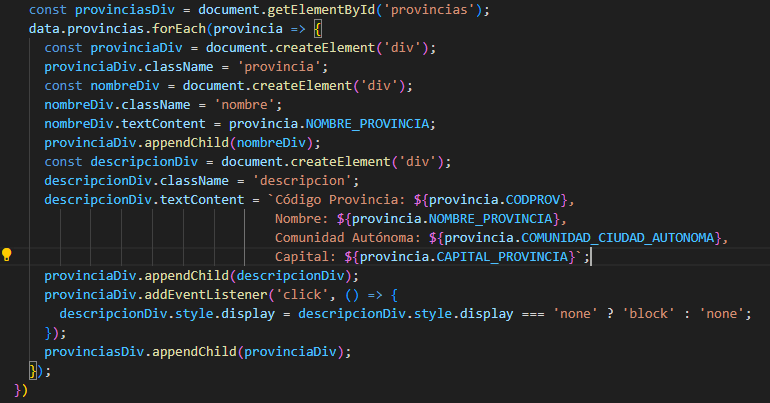


  
  
**fetch('https://www.el-tiempo.net/api/json/v2/provincias'):** Esto realiza una solicitud a la URL proporcionada, que es la API de El Tiempo para obtener información sobre las provincias de España.

**.then(response => response.json**()): Esto convierte la respuesta de la solicitud en formato JSON.

**.then(data => { ... }):** Este bloque maneja los datos que se obtienen de la API.

**.catch(error => console.error('Error:', error));:** Esto maneja cualquier error que pueda ocurrir durante la solicitud.

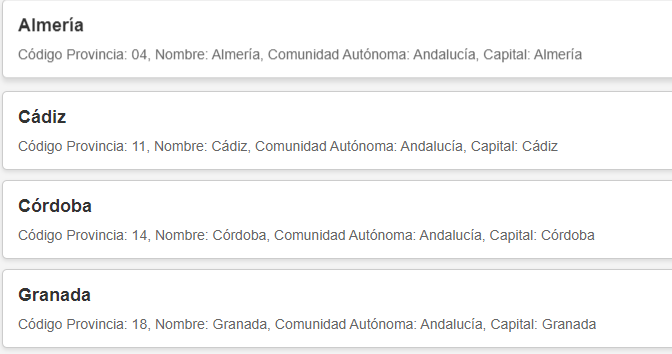


Este código recorre los datos de las provincias obtenidas de la API y crea un elemento de división (div) para cada provincia.

**const provinciasDiv = document.getElementById('provincias');:** Esto selecciona el elemento HTML con el ID "provincias" donde se mostrarán las provincias.

**data.provincias.forEach(provincia => { ... }):** Esto itera sobre cada provincia en los datos obtenidos.

Para cada provincia, se crea un nuevo elemento div llamado provinciaDiv y se le asigna la clase "provincia". Se crea un elemento div para el nombre de la provincia (nombreDiv) y se le asigna la clase "nombre". Se establece el contenido de texto del elemento nombreDiv como el nombre de la provincia. Se crea un elemento div para la descripción de la provincia (descripcionDiv) y se le asigna la clase "descripcion". Se establece el contenido de texto del elemento descripcionDiv con información detallada sobre la provincia, incluyendo su código, nombre, comunidad autónoma y capital. Se agrega un event listener al elemento provinciaDiv para que cuando se haga clic en él, se muestre o se oculte la descripción de la provincia.



**Conclusiones**

Ambos códigos muestran ejemplos prácticos de cómo utilizar tecnologías web para crear aplicaciones interactivas. La calculadora demuestra el uso de HTML, CSS y JavaScript para implementar una funcionalidad básica, mientras que la aplicación de la API muestra cómo obtener y mostrar datos de fuentes externas.