Calculator Javascript Basics

HTML tag or doctype html tag 🡪 Declares it is an HTML5 document.

It is the root element of HTML Document.

| **Tag** | **Meaning** |
| --- | --- |
| <!DOCTYPE html> | Declares HTML5 document |
| <html lang="en"> | Root of the HTML document, sets language to English |
| <head> | Contains metadata (not visible on page) |
| <meta charset="UTF-8"> | Sets character encoding to UTF-8 |
| <meta name="viewport"...> | Makes the site responsive for mobile |
| <title> | Sets the browser tab title |
| <body> | Contains all the visible content |

Class vs id

The class attribute is used to **group multiple elements** together under a common name so you can style them or manipulate them (with CSS or JavaScript).

The id attribute is used to uniquely identify **a single HTML element**.

Input tag :

Position Properties :

1. Static : Default type of an container
2. Relative : With respect to the window where you want to place top,left,right or bottom.
3. Absolute : First you have to give its parent container property as relative then you can give the container absoslte property.

**3. Is it required?**

* **No**, it’s not strictly required to give a parent position: relative for an absolutely‑positioned child to work.
* **But** if you don’t, the child will look up the tree for any positioned ancestor. If none are found, it will position itself relative to the page’s root (the viewport), which often isn’t what you want.

1. Sticky : wrt its parent container.
2. Fixed : wrt its window.

**4) position: sticky**

* **Your summary:** “wrt its parent container.”
* **More precisely:**
  1. A sticky element acts **like position: relative** until the viewport (or nearest scrollable ancestor) scrolls past a threshold you’ve set with top/bottom/left/right.
  2. Once that threshold is reached, it “sticks” and behaves **like position: fixed**, **but only within the bounds of its parent (or nearest scrolling container)**.
  3. When the parent’s bottom (or right) edge is reached, the sticky element will scroll away again—so it never escapes its container.

**5) position: fixed**

* **Your summary:** “wrt its window.”
* **More precisely:**
  1. A fixed element is **always removed from the document flow** and positioned **relative to the viewport** (the browser window).
  2. It does **not** move when the page scrolls.
  3. It does **not** care about any parent containers—its position is locked to the viewport’s edges.

In the body css

user-select : none is used to avoid selecting text as the blue line comes accidently as if you selected a text but with this property the selection can be avoided.

Javascript :

### 🤔 Why use let in your calculator project?

Because:

* You need to **store a changing value** (like updating the screen with each button click).
* let allows **reassignment** (just like var) but gives **better scoping**.

Watched a video of let,var,const 🡪 from youtube by Road Side coder

let a = …

now we will use ‘ document.querySelector ‘

document is an object in the dom that represents the entire HTML Page loaded in the browser

query selector() is a method that lets you select the first element that matches with the particular css selectors

querySelectorAll() is a JavaScript method that selects ***all* elements** that match a given **CSS selector**, not just the first one.

It returns a **NodeList** (which is like an array) of all matching elements.

This line grabs **all the buttons** that have the class btn — like the numbers and operators in your calculator — and stores them in a list called btn.

Then, we can loop over them:

js

CopyEdit

for (let item of btn) {

item.addEventListener('click', (e) => {

const btntext = e.target.innerText;

screen.value += btntext;

});

}

✅ This means: “For every button with class btn, when it's clicked, get its text and add it to the screen.”