Why do we need section tag?

The <section> tag in HTML is used to group related content together within a webpage. It helps in organizing the document into meaningful sections, making it easier to structure and style the content.

The <section> tag is best used when the content within it follows a common theme or topic, distinguishing it from other parts of the webpage.

<section>

<h2>About Us</h2>

<p>We are a company that values innovation and quality.</p>

</section>

<section>

<h2>Our Services</h2>

<p>We offer web development, SEO optimization, and UI/UX design.</p>

</section>

Reusable Styling: Sections can be styled individually using CSS.

Enhances Maintainability: Easier to modify or update specific sections of a webpage without affecting other parts.

min-height vs height

height: 100vh (Fixed Height)

Forces the element to be exactly 100% of the viewport height.

If the content inside the <div> exceeds the viewport height, it may overflow, causing scrolling issues.

min-height: 100vh (Flexible Height)

Ensures that the element is at least 100% of the viewport height.

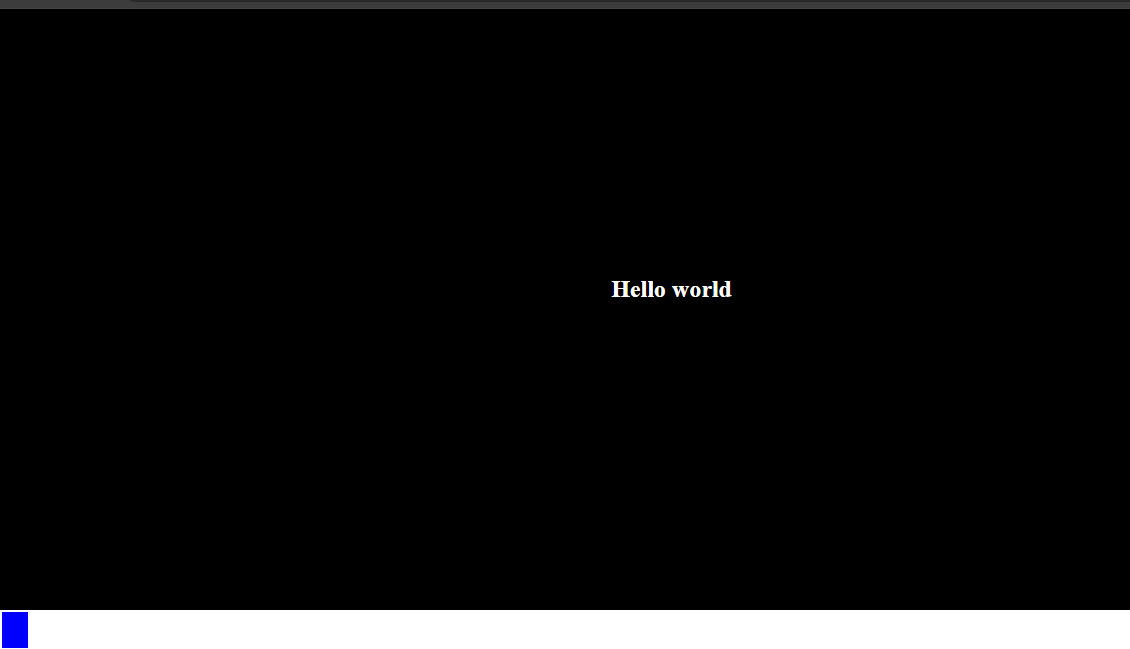
Allows the div to grow if the content inside it is larger than the viewport.

Prevents content from overflowing or being cut off unnecessarily.

Position: absolute

The position: absolute property removes an element from the normal document flow and positions it relative to the nearest positioned ancestor (i.e., an ancestor with position: relative, absolute, or fixed). If no positioned ancestor exists, it will be positioned relative to the <html> (or <body> in some cases).

If I would not have used above property , cursor div would have been in the normal document flow



The window.addEventListener('mousemove', (e) => {}) method is used to detect mouse movement across the entire window and execute a function whenever the mouse moves.

Event object

It is automatically passed to event listener callback functions.

You can use any variable name instead of e, such as a, d, c, or anything else. The event object is automatically implicitly passed as the first argument, so whatever variable you define as the first parameter will hold the event object.

If you pass an already defined variable as a parameter in the event listener callback, the variable's value will be overridden by the event object inside the function scope.

It contains detailed information about the event that occurred.

The properties of the event object depend on the event type.

For example:

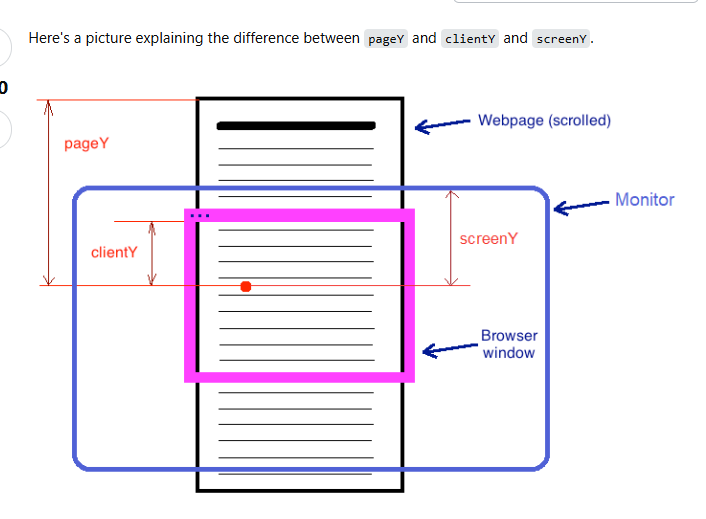
If the event type is 'mousemove', then event object contains mouse position relevant details (clientX, clientY,pageX ,pageY,screenX ,screenY ).

If the event is 'keydown', it contains key-related details (key, code, etc.).

If the event is 'click', it contains information about the clicked element.

Differences between

clientX, clientY,pageX ,pageY,screenX ,screenY



# clientX and clienty

These give the X and Y position of the mouse relative to the visible part of the webpage (the viewport).

They do not include scrolling. If you scroll down, clientY = 0 is still at the top of the visible area.

- Definition: The horizontal coordinate of the mouse pointer relative to the client area of the window.

- Client area: The area of the window excluding the browser’s title bar, borders, and scrollbars.

# screenX and screenY

- Values: 555, 87 (or any other non-zero values) at the top-left corner of the browser window.

- Reason: screenX and screenY provide the coordinates of the mouse pointer relative to the origin of the screen (the top-left corner of the physical screen).

- Why non-zero values?: The top-left corner of thebrowser window is not necessarily at the top-left corner of the physical screen , it might have been resized or The window might be positioned at a certain distance from the top and left edges of the screen, hence the non-zero values.

#pagex and pagey

pageX/Y coordinates are relative to the top left corner of the whole rendered page (including parts hidden by scrolling)

pageX/Y gives the coordinates relative to the <html> element in CSS pixels.

· These give the **X and Y position relative to the entire page**, including any scrolling.

· If you scroll **down**, pageY **increases**, but clientY stays the same

Why use px in following statement

cursor.style.top = a.pageY + 'px'

In CSS, sizes, positions, and spacing in a webpage must be specified with a unit, like px, %, em etc.

pageY gives a number (e.g., 200), but top in CSS requires a string with a unit (200px).

transform: translate(-30%, -20%);

The transform: translate(x, y); CSS property moves an element from its original position without affecting surrounding elements.

The transform property allows you to visually change an element by moving, rotating, scaling, or skewing it, without affecting the document flow or layout.

Move Elements without using position: absolute

translate(x, y) moves the element by x pixels/percentage to the right (or left if negative) and y pixels/percentage down (or up if negative).

-30% (X-axis) → Moves the element 30% to the left relative to its own width.

-20% (Y-axis) → Moves the element 20% upward relative to its own height.

When we apply the transform property, the element visually changes (moves, rotates, scales, etc.), but its original position in the document remains the same.

Using top (Affects Document Flow)

Top:5px

The element moves down by 50px, and other elements adjust accordingly.

window.scrollY

The window.scrollY property returns the number of pixels the document has been scrolled vertically.

cursor.setAttribute('data-fromTop', (cursor.offsetTop - scrollY ))

setAttribute is used to create a new or custom attribute for any element where in the 1st argument we pass attribute name and in the second we pass the value for attribute

cursor.getAttribute('data-fromTop')

To get the value of attribute of some element , we will pass the attribute name as argument

.cursor::after

The ::after is a CSS pseudo-element that allows you to insert content after an element’s content.

::after → A pseudo-element that adds extra content inside .cursor, but after its main content.

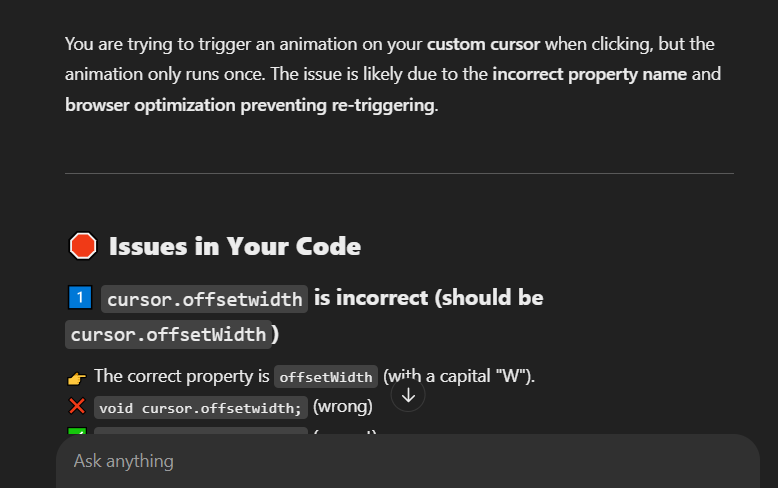
It does NOT create a separate element in the HTML, but behaves like one in CSS.

We can apply border radius property without1st applying border property

Noiceee

Just bcs of 1 letter , I have been messing up my mind from last night greatttttttt

You better keep this shit in your mind



@keyframes

is used to define which styles will be applied to an element at specific time intervals during an animation.

@keyframes name {

keyframes-selector {css-styles;}

keyframes-selector {css-styles;}

...

}

@keyframes mymove {

0% {top: 0px; background: red; width: 100px;}

100% {top: 200px; background: yellow; width: 300px;}

}

animation: click 0.5s ease-in-out forwards;