How to Determine the Middle of Cursor?

One of the most interesting aspects was that the tooltip appears in the middle of our cursor. How can we do that? Let's find out in this lesson.

WE'LL COVER THE FOLLOWING

- What we have so far
- Exploring solutions
- Employing devtools

What we have so far

So far, we've investigated if we can use constant character widths to determine the middle, and we've tried to use the `getSelection` method alone to provide all the information we need.

We know that we'll certainly have to use *some* information from `getSelection`, since it reliably gives us the selected characters. So the problem becomes this: given a string of characters in an HTML node, find the x and y offsets of the middle (so that we can have absolute position for the tooltip with `top` and `left` properties).

We can divide this into subproblems. If we can find the x and y offsets of *any* HTML element, one idea is to wrap the selected text in a span, get the x and y offsets and the width of that span, and consider the middle to be `(x + width/2, y)`.

Googling "get x and y of html node," I find the API `getBoundingClientRect()`, which has the properties we're looking for. I'm not entirely sure how to determine if the selection spans multiple lines. Maybe we can use the `height` to determine that. We'll see.

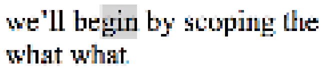
Let's write some code to test if this method works.

I’m using `console.log` to print often, and continuously test if my intermediary steps work, like getting the selected text.

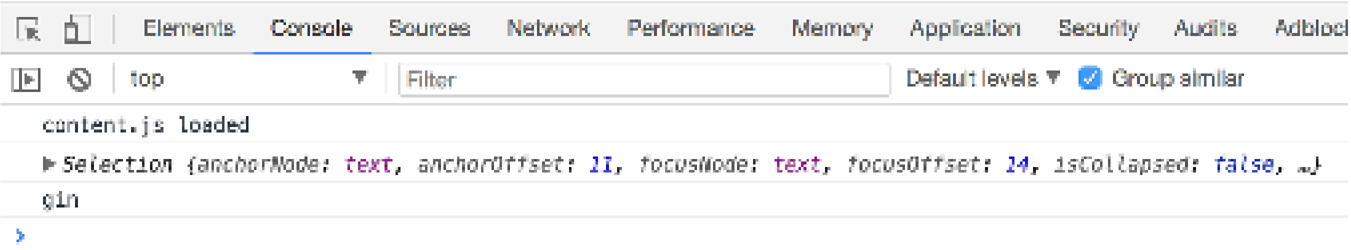
```
document.onmouseup = () => {
  const selection = document.getSelection();
  console.log(selection);
  const anchorNode = selection.anchorNode;
  const focusNode = selection.focusNode;

  if (anchorNode != focusNode) {
    // Cross-paragraph selection
    return;
  }

  const selectedText = anchorNode.data.substring(selection.anchorOffset, selection.focusOffse
  console.log(selectedText);
}
```



we'll begin by scoping the
what what



```
content.js loaded
▶ Selection {anchorNode: text, anchorOffset: 11, focusNode: text, focusOffset: 14, isCollapsed: false, …}
gin
>
```

The result is this:

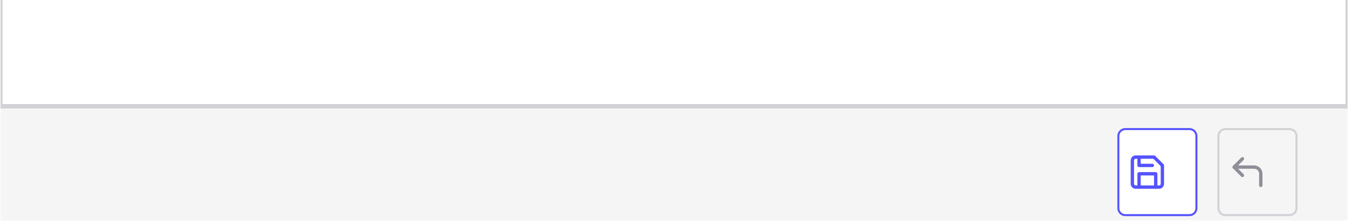
Output

JavaScript

HTML

CSS (SCSS)

we’ll begin by scoping the
what what



Try highlighting text. Beware, it only works once! If you want to play with it more, switching tabs (between, for example, the JavaScript and output) will cause a refresh.

Exploring solutions

I applied a CSS property to turn the class red for some visual confirmation of what I'm doing. The big issue now is that replacing `innerHTML` seems to remove my highlighted text. That's not acceptable. Maybe JavaScript can force a re-highlight of the original selection? Googling around, I can't find anything. Now we want to make sure that what Medium is doing isn't just applying a background color to appear highlighted. If that were the case, highlighting some text and doing a copy-paste operation should not work, yet it does on Medium.

At a higher level, it makes sense that innerHTML replacement removes my original highlight. The text that was highlighted is technically no more (even if the text output looks the same). They're not the same object. Inserting spans, however, means modifying the HTML. There doesn't seem to be a way to reconcile that the text highlight is maintained properly given this method.

Employing devtools

I'm curious if Medium is just doing character width estimations to get the middle, so I go into the `devtools` again and start removing properties related to font. I change the font-family to Arial, change the font size and character spacing. The tooltip seems to be able to adapt.

This method doesn’t seem to work. Let’s try something else!