

1. Reproduce the Bug

**DEMO: GET STARTED DEBUGGING  
JAVASCRIPT WITH CHROME DEVTOOLS**

**NUMBER 1**  **NUMBER 2**

a.

**DEMO: GET STARTED DEBUGGING  
JAVASCRIPT WITH CHROME DEVTOOLS**

**NUMBER 1**  **NUMBER 2**

b.

**5 + 6 = 56**

c.

- d. Added in two input fields A and B to represent two values. Rather than adding the numbers, they are being concatenated.

```

/*For demo work, demonstrate concatenation as addition*/
const label = document.querySelector('.bug-example');
const num1Input = document.getElementById('num1');
const num2Input = document.getElementById('num2');
const button = document.querySelector('.bug-example button');

// Add click event to the button
button.addEventListener('click', onClick);

// Functions to get the input values
function getNumber1() {
    return num1Input.value;
}

function getNumber2() {
    return num2Input.value;
}

// Logic taken from google dev tools demo: https://developer.chrome.com/docs/devtools/javascript
function onClick() {
    if (inputsAreEmpty()) {
        label.textContent = "Please enter values in both input fields.";
        return;
    }
    updateLabel()
}

function inputsAreEmpty() {
    if (getNumber1() === "" || getNumber2() === "") {
        return true;
    } else {
        return false;
    }
}

function updateLabel() {
    const addend1 = getNumber1();
    console.log('addend1:', addend1);
    const addend2 = getNumber2();
    console.log('addend2:', addend2);
    const sum = addend1 + addend2;
    console.log('sum:', sum);
    label.textContent = addend1 + ' + ' + addend2 + ' = ' + sum;
}

```

You, 4 seconds ago • Uncommitted changes

i.

```

/*For demo work */
bug-example {
    text-align: center;
    margin-top: 50px;
    padding: 30px;
    background-color: white;
    border-radius: 15px;
    max-width: 600px;
    margin-left: auto;
    margin-right: auto;
}

```

ii.

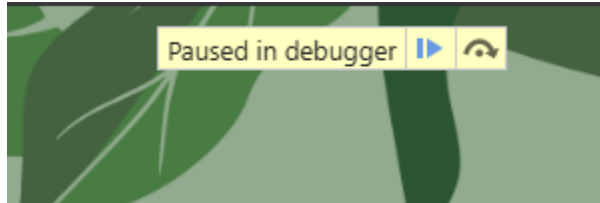
```

<div class="bug-example">
  <h2>Demo: Get Started Debugging JavaScript with Chrome DevTools</h2>
  <label for="num1">Number 1</label>
  <input placeholder="Number 1" id="num1">
  <label for="num2">Number 2</label>
  <input placeholder="Number 2" id="num2">
  <button>Add Number 1 and Number 2</button>
</div>

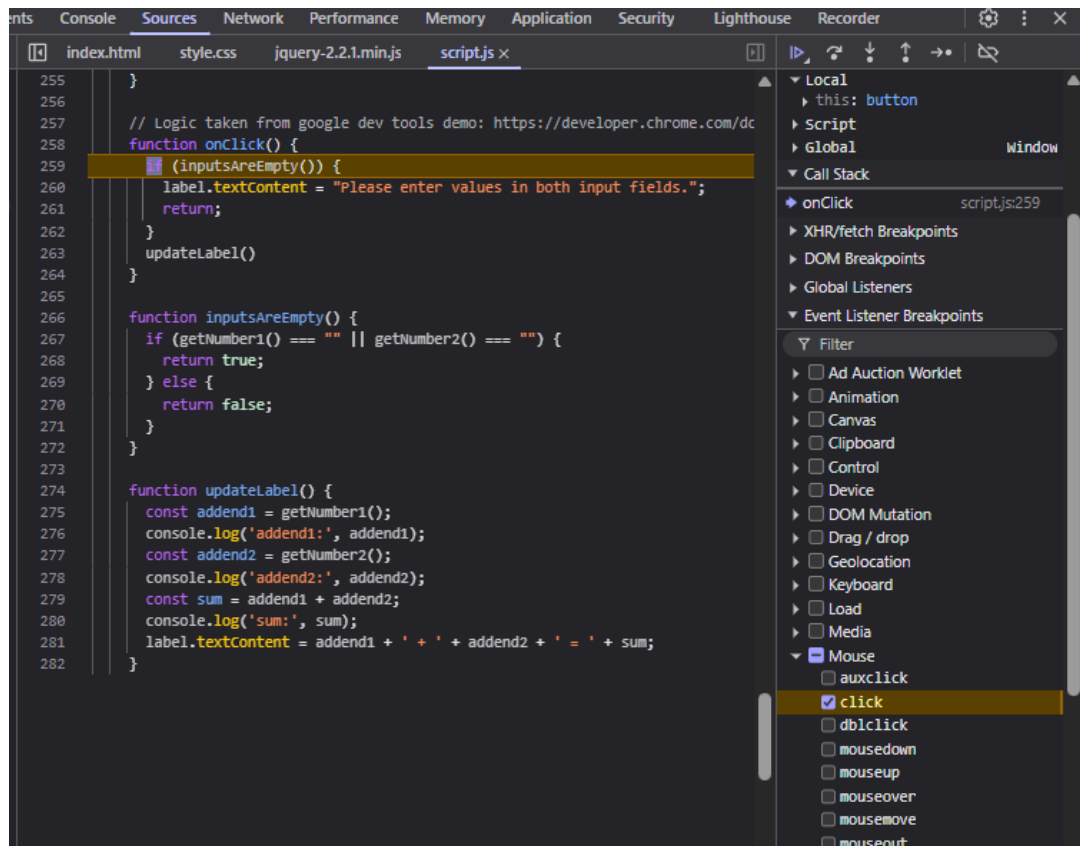
```

iii.

2. Pause the code, Step through code, Set a line-of-code breakpoint,



a.



b.

```
254     return num2Input.value;
255 }
256
257 // Logic taken from google dev tools demo: https://developer.chrome.com/dc
258 function onClick() {
259     if (inputsAreEmpty()) {
260         label.textContent = "Please enter values in both input fields.";
261         return;
262     }
263     updateLabel()
264 }
265
266 function inputsAreEmpty() {
267     if (getNumber1() === "" || getNumber2() === "") {
268         return true;
269     } else {
270         return false;
271     }
272 }
273
274 function updateLabel() {
275     const addend1 = getNumber1(); addend1 = "7"
276     console.log('addend1:', addend1);
277     const addend2 = getNumber2(); addend2 = "7"
278     console.log('addend2:', addend2);
279     const sum = addend1 + addend2; sum = "77", addend1 = "7"
280     console.log('sum:', sum);
281     label.textContent = addend1 + ' + ' + addend2 + ' = ' + sum;
282 }
```

c.

```
addend1: 7
addend2: 7
sum: 77
```

d.

e. Code was paused, breakpoint added using event listener. We needed to move through the various functions by stepping into and stepping over the function calls.

3. Check variable values, (Method 1, 2,3)

```
Local
  this: Window
  addend1: "8"
  addend2: "8"
  sum: "88"
```

a.

b. Method 1 was inspecting the scope. In viewing my local vars, we see the different values for the addends and notice how they are wrapped in quotes- an indicator that our addition may be string concatenation.

```
Watch
  typeof sum: "string"
```

c.

- d. Method 2 utilizes the 'watch' feature. Here we see that our sum variable is actually a string.

```
> parseInt(addend1) + parseInt(addend2)
< 16
```

- e.
- f. Method 3 involves the use of the console. If we parse the addends as ints, we see that the correct value is returned when added together (8 + 8 is 16 rather than 88).

4. Apply a fix

a. **4 + 6 = 10**

- b. 

```
const sum = parseInt(addend1) + parseInt(addend2);
```
- c. Added fix by parsing in the addends as integers.

5. Examples From (View DOM nodesLinks to an external site->  
<https://developer.chrome.com/docs/devtools/dom>)

- a. Examples done below!

6. View DOM nodes (all sub parts)

```
<!--Example with painters-->
<ul>
  <li>
    ::marker
    "Michelangelo"
  </li>
  <li>⋮</li>
</ul>
<!--Example with the cities-->
<ul>
  <li>
    ::marker
    "Tokyo" == $0
  </li>
  <li>⋮</li>
</ul>
</div>
```

a.

```
--Examples and code from DOM demo-https://developer.chrome.com/docs/devtools/dom
<div class>
  <!--Example with painters-->
  <ul>
    <li>Michelangelo</li>
    <li>Raphael
      The <strong>Elements</strong> panel of DevTools opens.
    <code translate="no" dir="ltr">&lt;li&gt;Michelangelo&lt;/li&gt;</code>
  </ul>
  <!--Example with the cities-->
  <ul>
    <li>Tokyo</li>
    <li>Beirut</li>
  </ul>
</div>
```

b.

- c. Added in the same lists from the tutorial. Inspected Michelangelo and Tokyo.

```
</div>
<!--Example with names-->
<ul>
  <li>
    ::marker
    "George"
  </li>
  <li>
    ::marker
    "Ringo"
  </li>
  <li> == $0
    ::marker
    "Paul"
  </li>
  <li>
    ::marker
    "John"
  </li>
</ul>
```

- d.
- e. Using arrow keys to navigate through the list and open them up. Up/down arrow to navigate through the items, right/left to open and close the items themselves.

## DEMO: GET STARTED DEBUGGING JAVASCRIPT WITH CHROME DEVTOOLS

NUMBER 1

NUMBER 2

Add Number 1 and Number 2

- MICHELANGELO
- RAFAEL THE ELEMENTS PANEL OF DEVTOOLS OPENS.  
<li>Michelangelo</li> IS HIGHLIGHTED IN THE DOM TREE.
- TOKYO
- BEAUF
- GEORGE
- ALNO
- PAUL
- JOHN

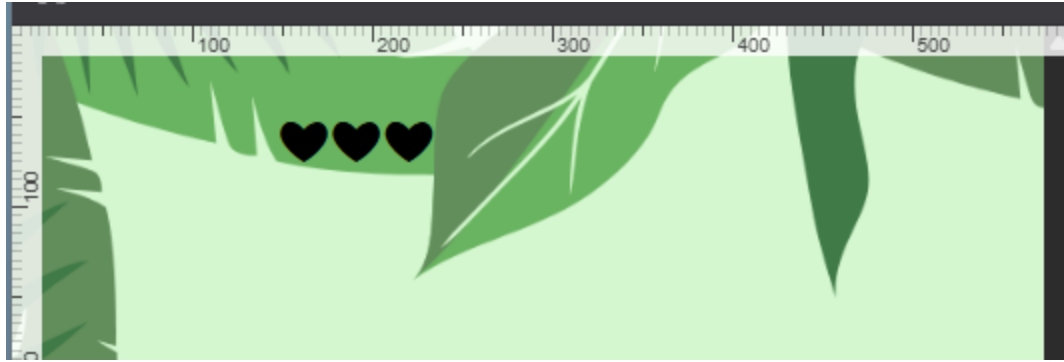
1. RIGHT-CLICK MAGRITTE BELOW AND SELECT INSPECT.

- MAGRITTE
- SOUTINE

2. SCROLL TO THE APPENDIX SECTION SCROLL INTO VIEW FOR ADDITIONAL INSTRUCTIONS.

```
<!--For creation of visual notif after each bu
<div class="pet-notif"></div>
<div class="button-container"></div> flex
</main>
<!--Layout taken from google dev tools demo: htt
chrome.com/docs/devtools/javascript-->
<div class="bug-example"></div>
<!--Examples and code from DOM demo-https://deve
m/docs/devtools/dom -->
<div class"></div>
<!--Example with names-->
<ul></ul>
<!--Example dealing with viewport-->
<ol>
  <li>
    ::marker
    <p></p>
    <ul>
      <li>
        ::marker
        "Magritte" == $0
      </li>
      <li>
        ::marker
        "Soutine"
      </li>
    </ul>
  </li>
  <li></li>
</ol>
<!-- Your web-app is https, so your scripts need
<script src="https://code.jquery.com/jquery-2.2.
html body ol li ul li (text)
Styles Computed Layout Event Listeners DOM Brea
eip6963:requestProvider
wallet-standard:app-ready
```

- f.
- g. Scroll into view to take user back to point of interest.



h.

i. Addition of show rulers via command ctrl+shift+p.

```

    <div>
      "Name: "
      <strong>
        <span class="name">Tigress</span>
      </strong>
    </div>
  <div>
    "Weight: " == $0
    <strong></strong>
  </div>
</div>
<!--Adding another button to represent another activity-->
<div></div>
</section>
</div>
<!--For creation of visual notif after each button press-->
<div class="pet-notif"></div>
<div class="button-container"></div>
</main>
<!--Layout taken from google dev tools demo: https://developer.chrome.
com/docs/devtools/javascript-->

```

j.

k. Search feature. Searched for 'weight', which was an important variable in my project 2.

7. Edit the DOMLinks to an external site. (Edit the content, Edit attributes, Edit node type, Edit HTML)

a. Edit the content

```
</li>
<li>
  ::marker
  "Leela" == $0
</li>
</ul>
```

i.

```
<!--Example for editing from devtools.-->
<ul> You, 1 second ago • Uncommitted changes
  <li>Fry</li>
  <li>Michelle</li>
</ul>
```



ii.

iii. I added the sample list into my file and then used the devtools to directly edit the element from Michelle to Leela.

b. Edit Attributes



i.

```
<ul>
  <li style="background-color:gold">
    ::marker
    "Howard"
  </li>
  ...
</ul>
```

ii.

iii. Added in a new attribute to the list element to make the background gold.

c. Edit Node Type



i.

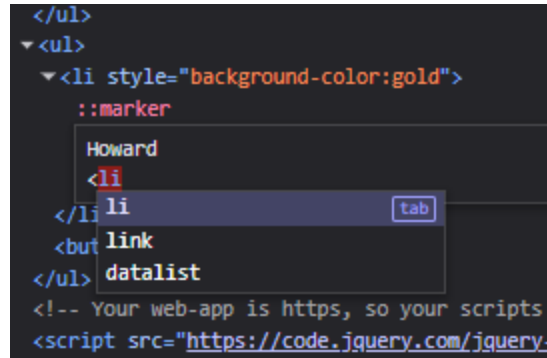
```
</li>
<button>Vince</button> == $0
</ul>
```

ii.

iii. Change the li element to a button element.

d. Edit HTML

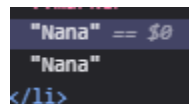




- i.
- ii. Editing the elements as html using autocomplete within the dev tools.

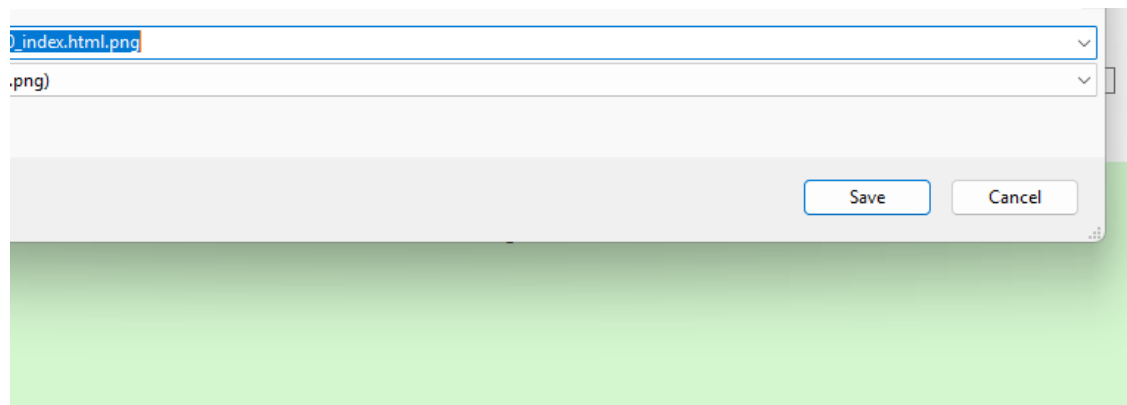
8. (Duplicate a node, Capture a screenshot, Reorder DOM nodes, Force state, Hide a node, Delete a node)

a. Duplicate a Node

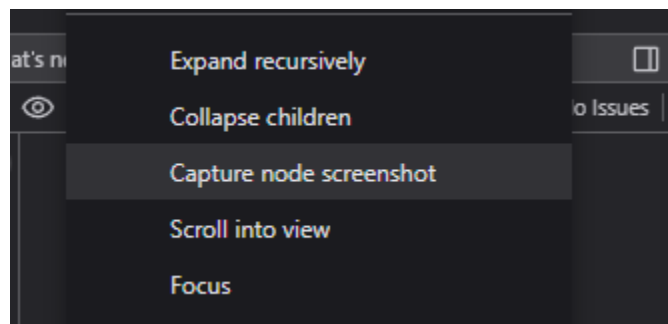


- i.
- ii. Right click to duplicate a value in the test list.

b. Capture a Screenshot

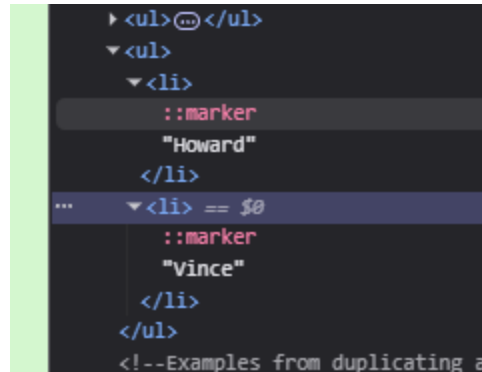


i.



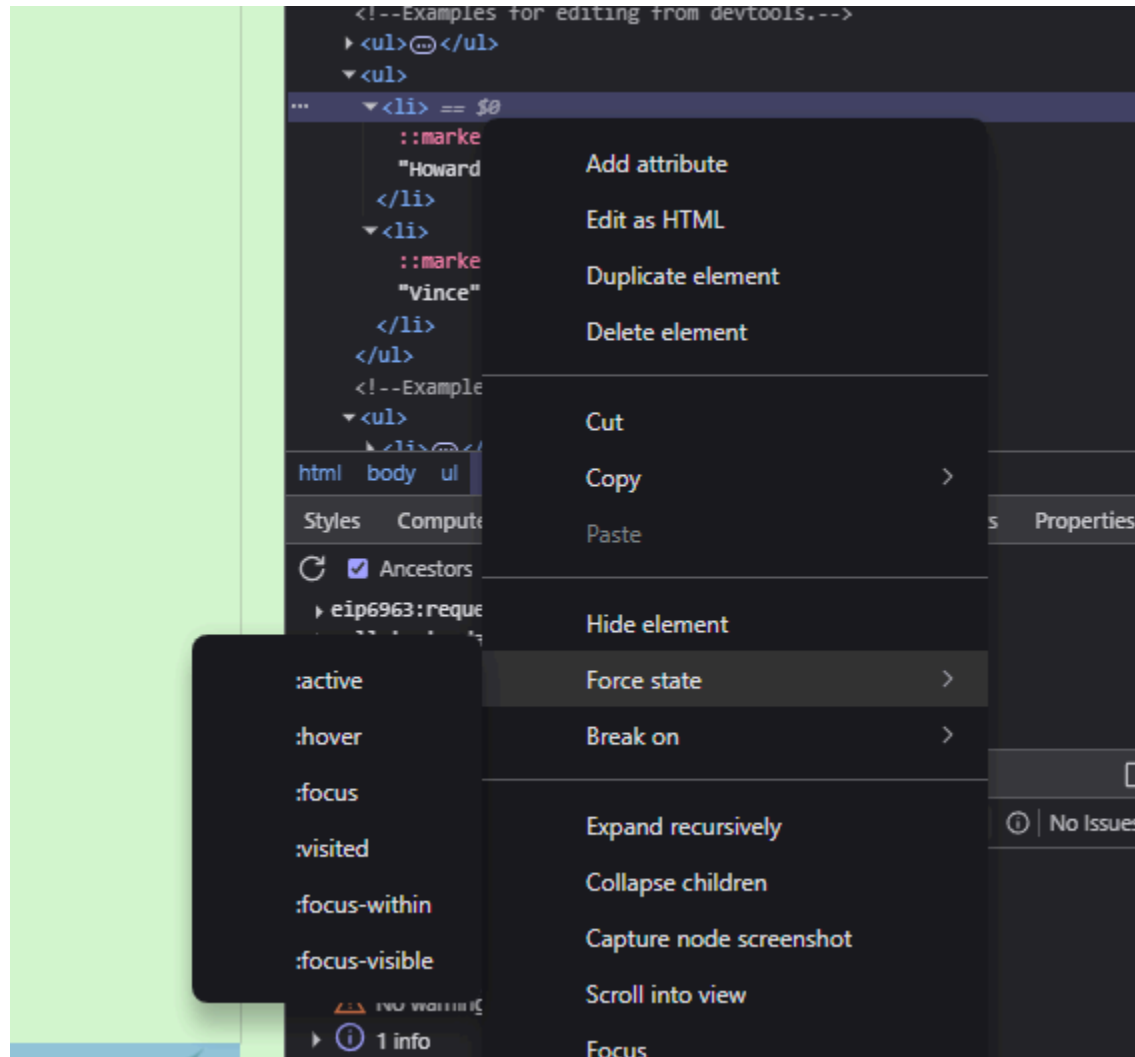
- ii.
- iii. Capture node screenshot by right clicking and selecting the above option.

c. Reorder DOM nodes



- i.
- ii. Hold and drag the varying elements to change their order. I added the lists from the demos to my project, opened them with live server, opened dev tools on live server, and reordered them like so.

d. Force State

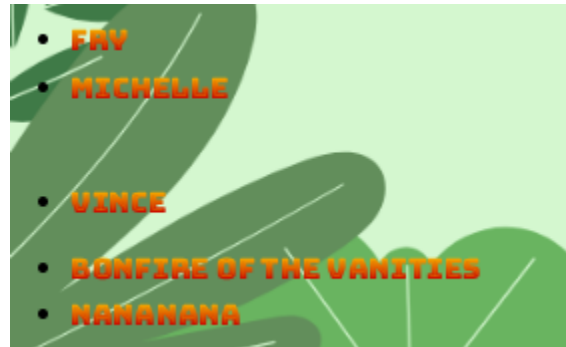


- i.
- ii. Select any of the states from the dropdown for the nodes.

e. Hide a Node

```
</ul></ul>
<ul>
  <li class="__web-inspector-hide-shortcut__" == $0
    ::marker
    "Howard"
  </li>
  <li>
    ::marker
```

i.



ii.

iii. With the sample lists, I hid the list element containing Howard.

f. Delete a Node

```
<!--Examples for editing from dev
<ul></ul>
<ul>
  <li == $0
    ::marker
    "Vince"
  </li>
</ul>
```

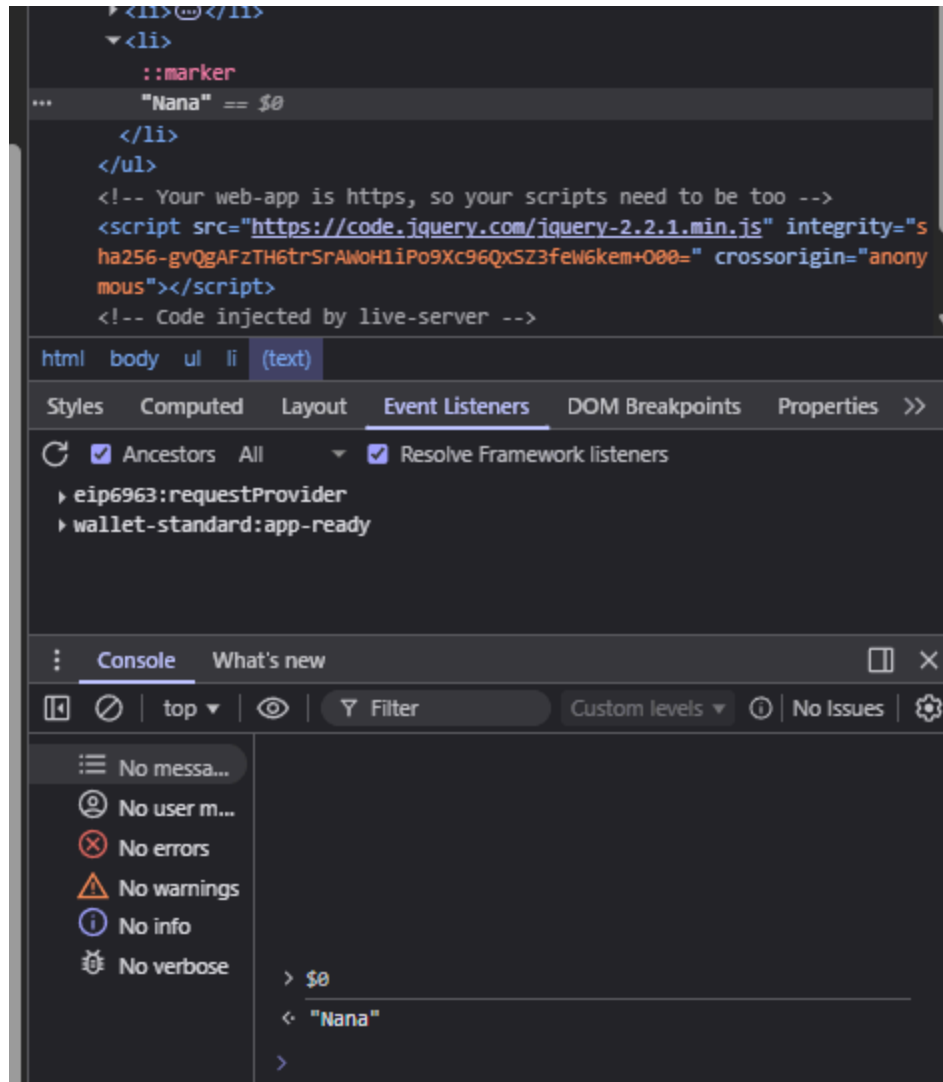
i.



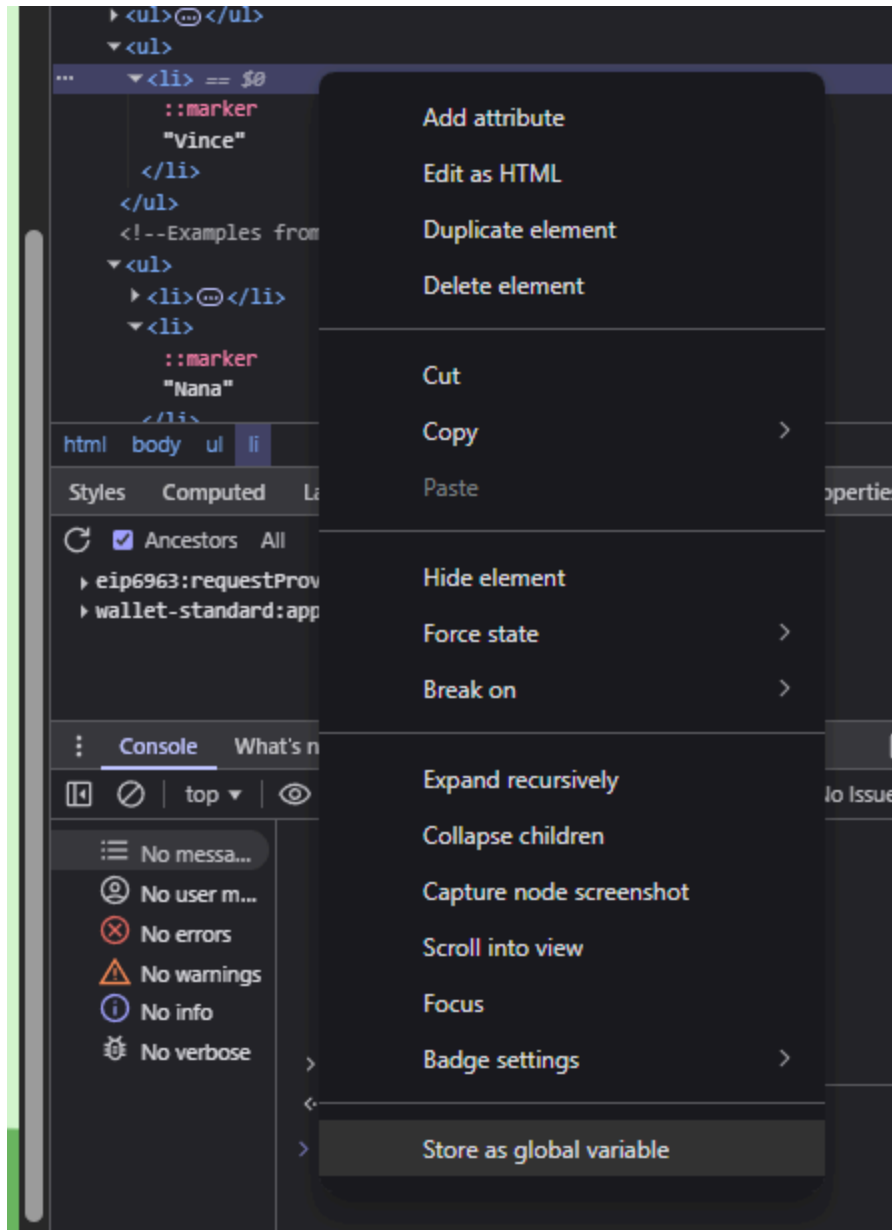
ii.

iii. The Howard element is now deleted.

9. Access Nodes in the Console



- a.
- b. For this example, I added the various lists into my project. From there we can open up the project in live server, open dev tools, and select on a node. Selecting the node gives that `'== $0'` value, and typing in `$0` into the console allows us to access that node.

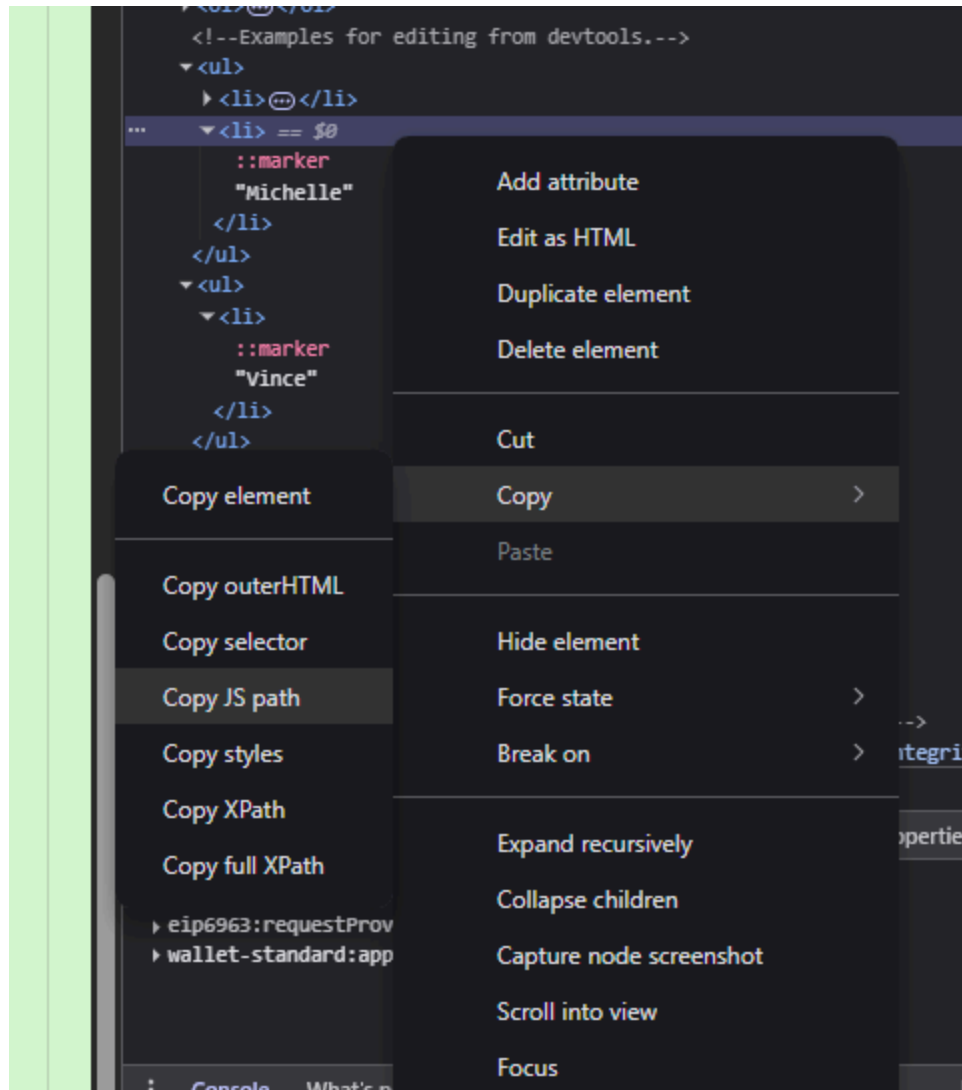


c.

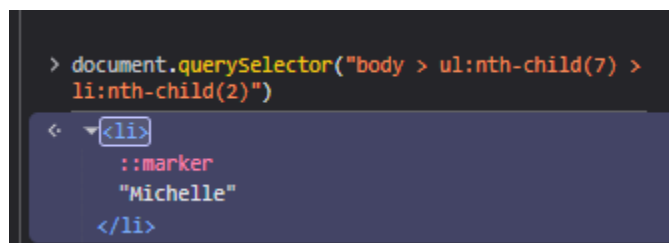


d.

- e. We are also able to store nodes as global variables. Here, I selected a node from one of the sample lists I added. I stored the list element containing 'Vince' as a global variable, and now we are able to access it in the console as 'temp 1'.



f.



g.

- h. For this example, I took one of the lists I copied over and copied the js path. From there, I copy pasted the path into the console. Using the JS path, we are able to retrieve the node.