Teaching Presentation for CIT 261-week 5

DOM MANIPULATION

- Using createElement
- appendChild
- insertBefore
- removeChild

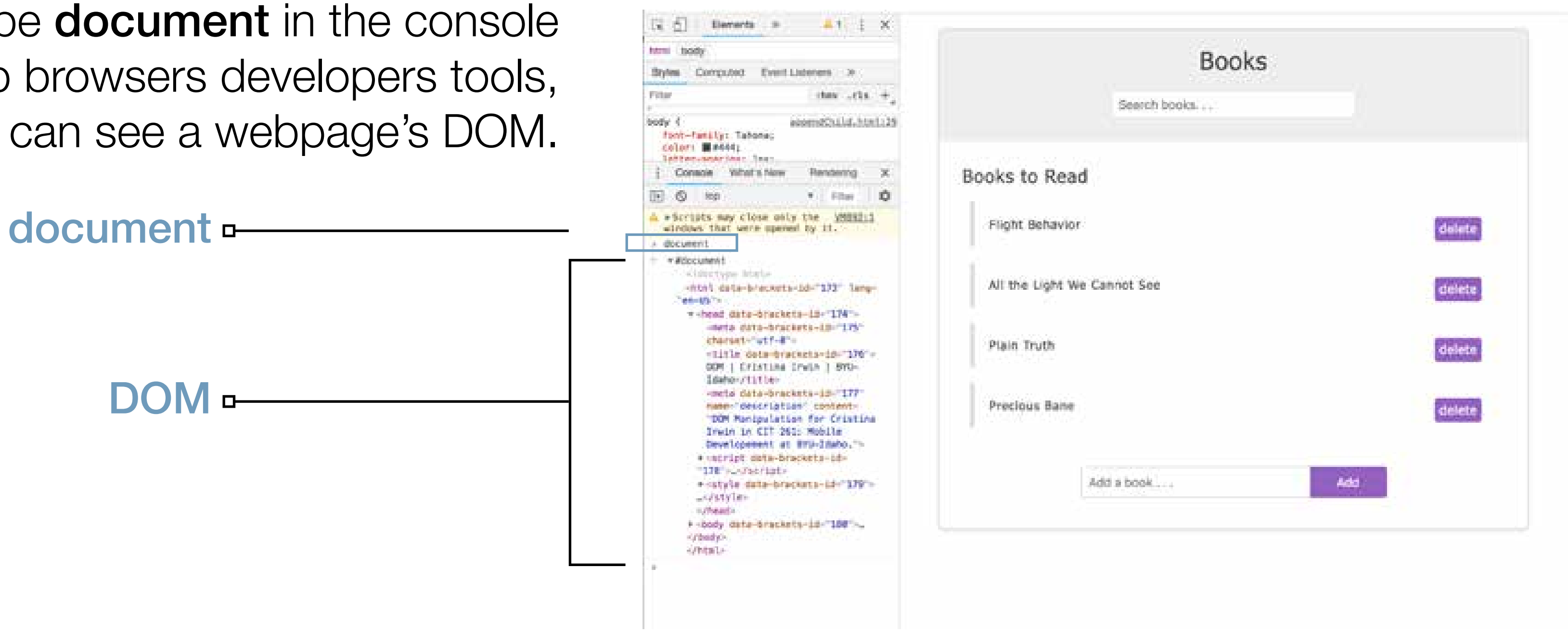
Cristina Irwin

DOM-What is it?

DOM stands for Document Object Model. DOM is created by the browser when a webpage is loaded. It is a graphical form, a tree of elements/nodes. You can use JavaScript to change the DOM. You can change or remove HTML elements, change and add Css Styles to an element, read and change element attributes (href, src, alt), create new HTML elements and insert them in the DOM, and attach event listeners to elements (click and submit).

w3schools and NetNinja

If you type **document** in the console in a web browsers developers tools, you can see a webpage's DOM.

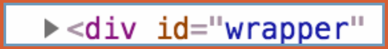


```
body {                                   appendChild.html:29
  font-family: Tahoma;
  color: ■#444;
  letter-spacing: 1px;
```

```
⚠ ▸Scripts may close only the    VM892:1
  windows that were opened by it.
> document
< ▾#document
    <!doctype html>
    <html data-brackets-id="173" lang=
    "en-US">
      ▾<head data-brackets-id="174">
         <meta data-brackets-id="175"
         charset="utf-8">
         <title data-brackets-id="176">
         DOM | Cristina Irwin | BYU-
         Idaho</title>
         <meta data-brackets-id="177"
         name="description" content=
         "DOM Manipulation for Cristina
         Irwin in CIT 261: Mobile
         Developement at BYU-Idaho.">
       ▸<script data-brackets-id=
       "178">…</script>
       ▸<style data-brackets-id="179">
       …</style>
       </head>
     ▸<body data-brackets-id="180">…
     </body>
    </html>
>
```

Access DOM Element

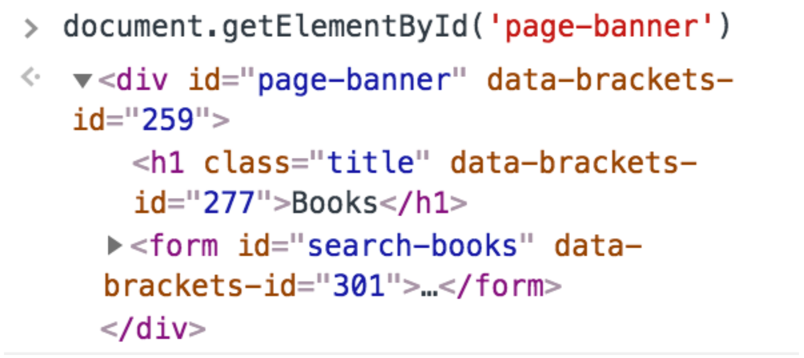
You use id's to access DOM elements.



```
▶<div id="wrapper"
```

Id's need to be unique.

Element/Node

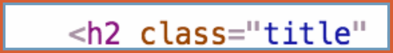


```
> document.getElementById('page-banner')
<· ▼<div id="page-banner" data-brackets-
  id="259">
      <h1 class="title" data-brackets-
      id="277">Books</h1>
    ▶<form id="search-books" data-
    brackets-id="301">…</form>
    </div>
```

In the console, type **document.getElementById('id-name')** and hit return.
The element/node will be shown in the console.

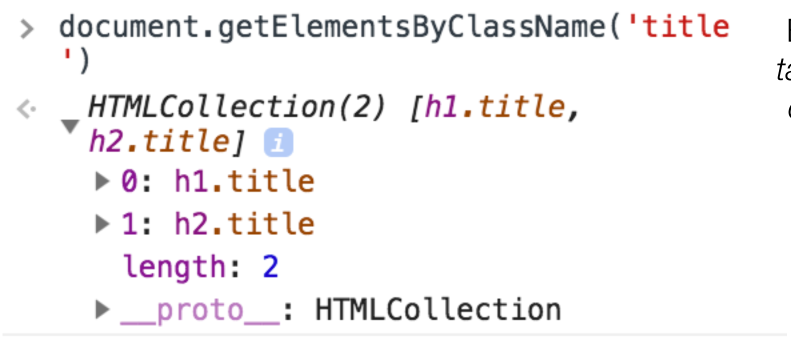
Access DOM Element

You can use class names to access DOM elements. Class names can be used multiple times. When use class name, you will get multiple items.



```
<h2 class="title"
```

Element/Node



```
> document.getElementsByClassName('title
  ')
<· ▼HTMLCollection(2) [h1.title,
    h2.title] i
    ▶0: h1.title
    ▶1: h2.title
     length: 2
    ▶__proto__: HTMLCollection
```

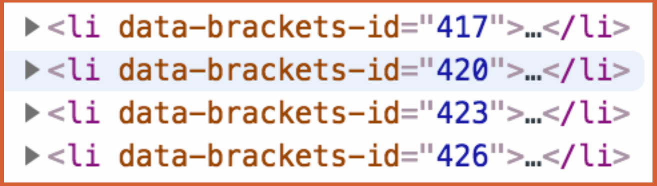
What is a Node?

Everything in a document is a node *(html tag, head tag, body tag, div tag, text on a document, comment on a document, an attribute)*. It is important to understand what a node is because that is how the **DOM** is manipulated. Node types have numbers (https://www.w3schools.com/jsref/prop_node_nodetype.asp)

In the console, type document.getElementsByClassName('class-name') and hit return. The element/node will be shown in the console.

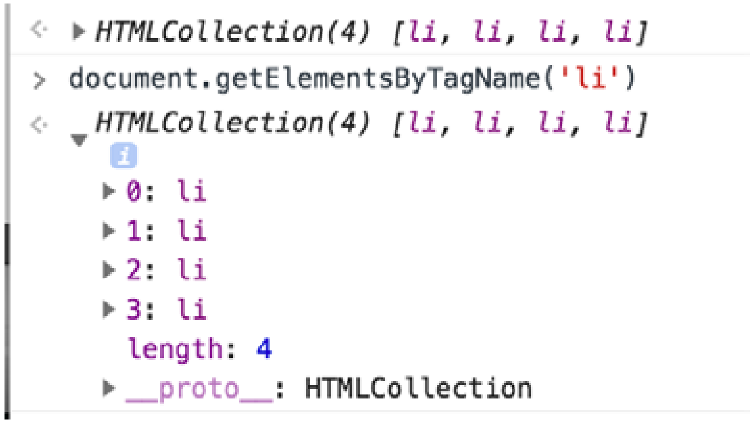
Access DOM Element

You can use tag name to access DOM elements. Tag name will return a collection of items that all have the same tag.



```
▶<li data-brackets-id="417">…</li>
▶<li data-brackets-id="420">…</li>
▶<li data-brackets-id="423">…</li>
▶<li data-brackets-id="426">…</li>
```

Element/Node



```
<· ▶HTMLCollection(4) [li, li, li, li]
>  document.getElementsByTagName('li')
<· ▼HTMLCollection(4) [li, li, li, li] i
      ▶0: li
      ▶1: li
      ▶2: li
      ▶3: li
       length: 4
      ▶__proto__: HTMLCollection
```

In the console, type document.getElementsByTagName(‘tag-name‘) and hit return. The element/node will be shown in the console.

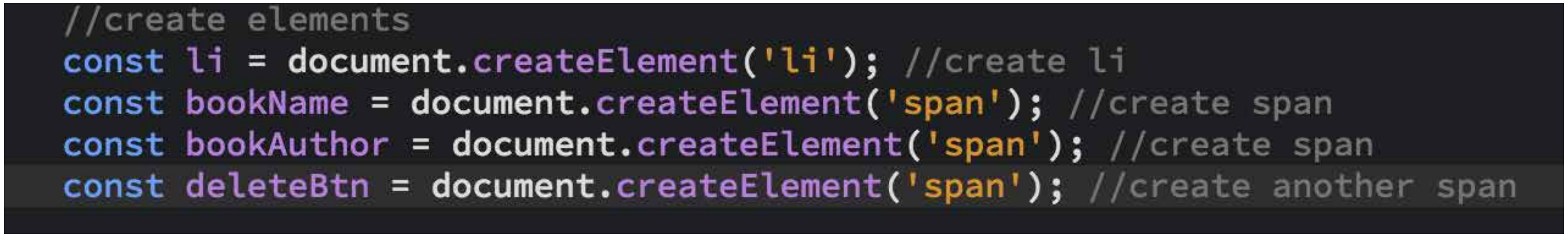
Traversing the DOM

To traverse the DOM upwards, use **varName.parentNode** `(list.parentNode)` and **var-Name.parentElement** `(list.parentElement)`. To Traverse the DOM downward, use **varName.children** `(list.children)`. To traverse on the same level (sibling to sibling) use **varName.nextSibling** `(list.nextSibling)`, **varName.nextElementSibling** `(list.nextElementSibling)`, **varName.previousSibling,** or **varName.previousElementSibling**. If null comes up, that means there is no “whatever you are calling”.

DOM Manipulation

.createElement

createElement means creating a new element on the DOM with a specific name. Once a new element has been created, other elements like appendChild, insertBefore, removeChild and many other methods can be used. W3Schools has a list of methods https://www.w3schools.com/jsref/dom_obj_document.asp



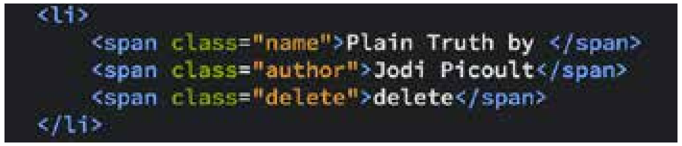
```
//create elements
const li = document.createElement('li'); //create li
const bookName = document.createElement('span'); //create span
const bookAuthor = document.createElement('span'); //create span
const deleteBtn = document.createElement('span'); //create another span
```

DOM Manipulation

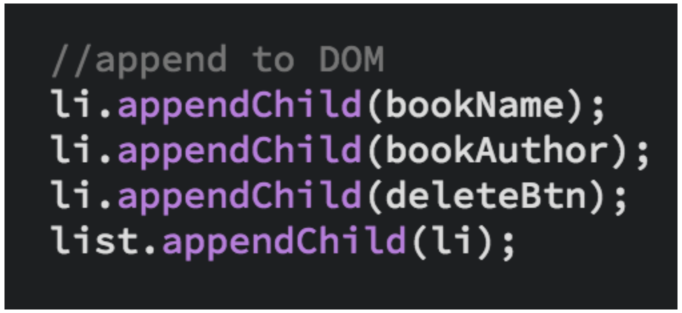
.appendChild

Append means “add”. appendChild means you add and an element(child) to a parent on the DOM.

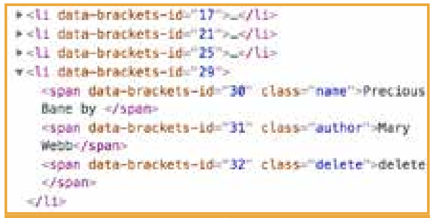
The child element will be added to the <li> parent. In this case, the child element is a <span>. appendChild will add a new span to the DOM



```
<li>
    <span class="name">Plain Truth by </span>
    <span class="author">Jodi Picoult</span>
    <span class="delete">delete</span>
</li>
```

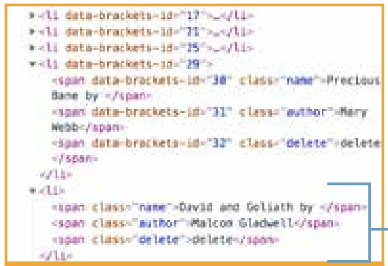


```
//append to DOM
li.appendChild(bookName);
li.appendChild(bookAuthor);
li.appendChild(deleteBtn);
list.appendChild(li);
```



```
▸ <li data-brackets-id="17">…</li>
▸ <li data-brackets-id="21">…</li>
▸ <li data-brackets-id="25">…</li>
▾ <li data-brackets-id="29">
    <span data-brackets-id="30" class="name">Precious
    Bane by </span>
    <span data-brackets-id="31" class="author">Mary
    Webb</span>
    <span data-brackets-id="32" class="delete">delete
    </span>
  </li>
```

BEFORE



```
▸ <li data-brackets-id="17">…</li>
▸ <li data-brackets-id="21">…</li>
▸ <li data-brackets-id="25">…</li>
▾ <li data-brackets-id="29">
    <span data-brackets-id="30" class="name">Precious
    Bane by </span>
    <span data-brackets-id="31" class="author">Mary
    Webb</span>
    <span data-brackets-id="32" class="delete">delete
    </span>
  </li>
▾ <li>
    <span class="name">David and Goliath by </span>
    <span class="author">Malcom Gladwell</span>
    <span class="delete">delete</span>
  </li>
```

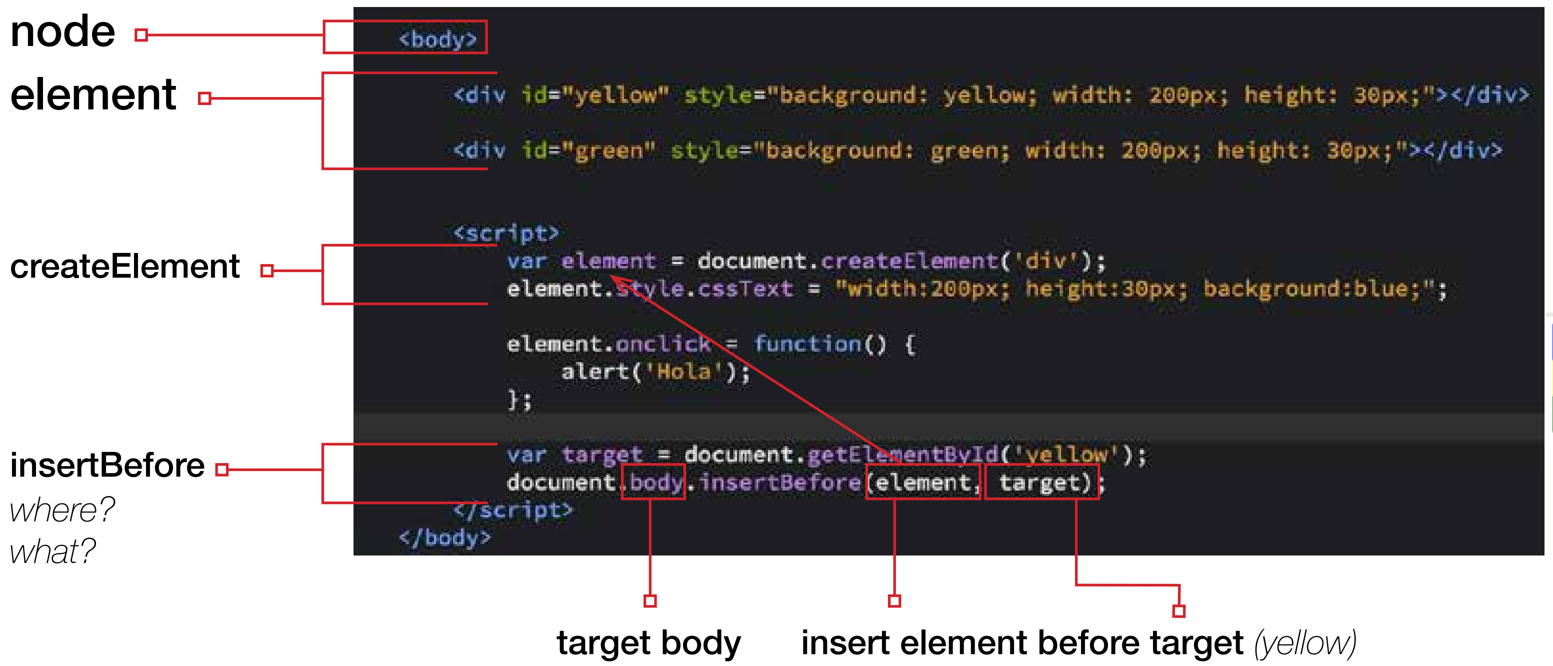
AFTER

new <li>, with new child elements, three <span>

DOM Manipulation

.insertBefore

Target a **node,** for example <body>. The element you are inserting will be inserted in that node. Target an element within the <body> node, and insert the element you wish to create.



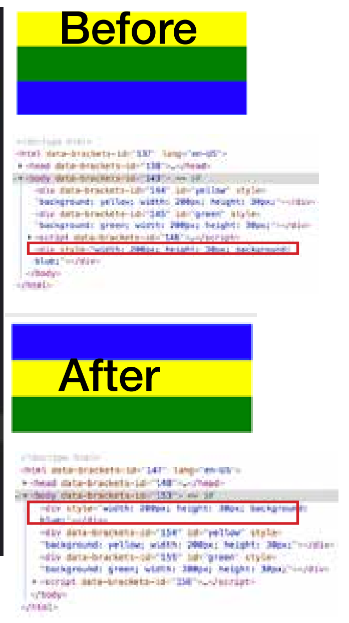
```
<body>
    <div id="yellow" style="background: yellow; width: 200px; height: 30px;"></div>

    <div id="green" style="background: green; width: 200px; height: 30px;"></div>


    <script>
        var element = document.createElement('div');
        element.style.cssText = "width:200px; height:30px; background:blue;";

        element.onclick = function() {
            alert('Hola');
        };

        var target = document.getElementById('yellow');
        document.body.insertBefore(element, target);
    </script>
</body>
```



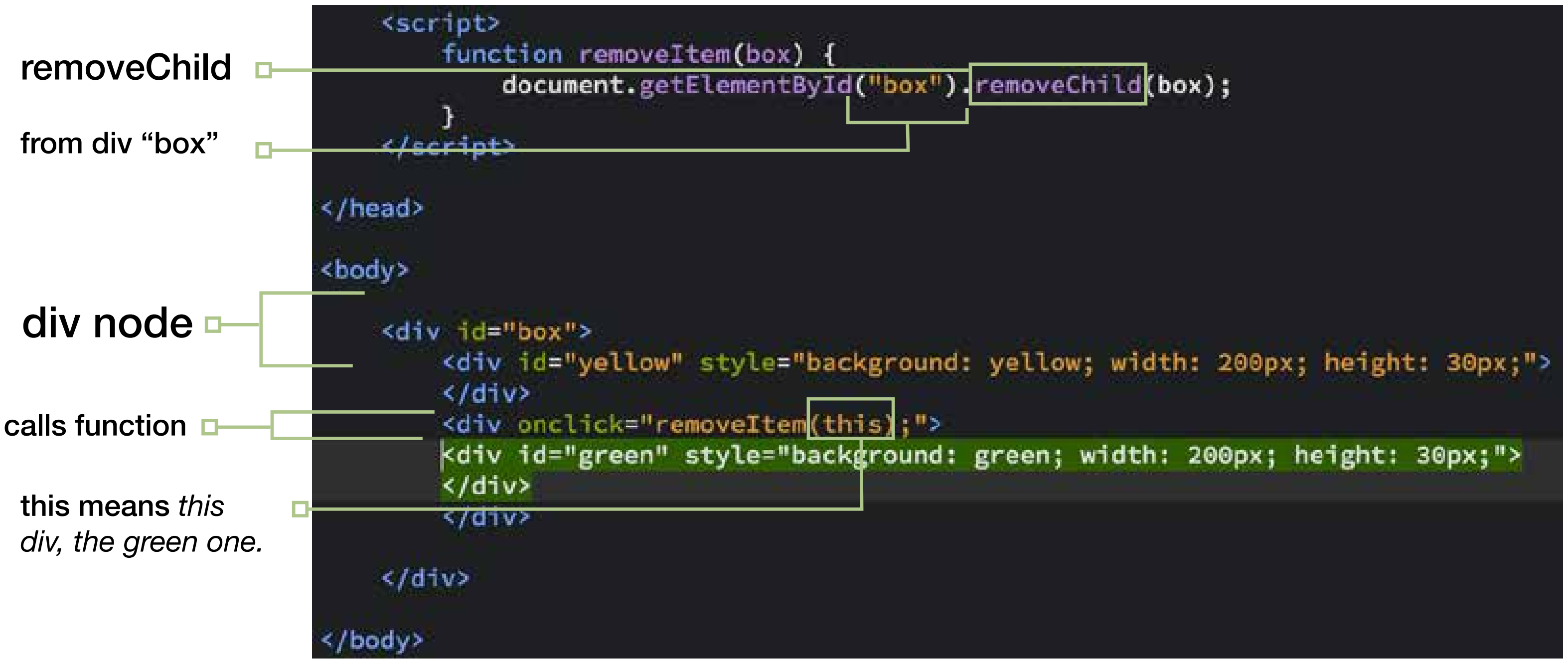
```
<!doctype html>
<html data-brackets-id="137" lang="en-US">
▶<head data-brackets-id="138">…</head>
▼<body data-brackets-id="143"> == $0
    <div data-brackets-id="144" id="yellow" style=
    "background: yellow; width: 200px; height: 30px;"></div>
    <div data-brackets-id="145" id="green" style=
    "background: green; width: 200px; height: 30px;"></div>
  ▶<script data-brackets-id="148">…</script>
    <div style="width: 200px; height: 30px; background:
    blue;"></div>
  </body>
</html>
```

```
<!doctype html>
<html data-brackets-id="147" lang="en-US">
▶<head data-brackets-id="148">…</head>
▼<body data-brackets-id="153"> == $0
    <div style="width: 200px; height: 30px; background:
    blue;"></div>
    <div data-brackets-id="154" id="yellow" style=
    "background: yellow; width: 200px; height: 30px;"></div>
    <div data-brackets-id="155" id="green" style=
    "background: green; width: 200px; height: 30px;"></div>
  ▶<script data-brackets-id="158">…</script>
  </body>
</html>
```

DOM Manipulation

.removeChild

removeChild removes a specific node from a specified element.. Once removed, it is no longer part of the DOM. **It is possible to store the removed child in a variable.**



```
    <script>
        function removeItem(box) {
            document.getElementById("box").removeChild(box);
        }
    </script>

</head>

<body>

    <div id="box">
        <div id="yellow" style="background: yellow; width: 200px; height: 30px;">
        </div>
        <div onclick="removeItem(this);">
        <div id="green" style="background: green; width: 200px; height: 30px;">
        </div>
        </div>

    </div>

</body>
```