

DOM - Document Object Model

By CODEMIND Technology

Contact us 966 5044 698
966 5044 598

JavaScript

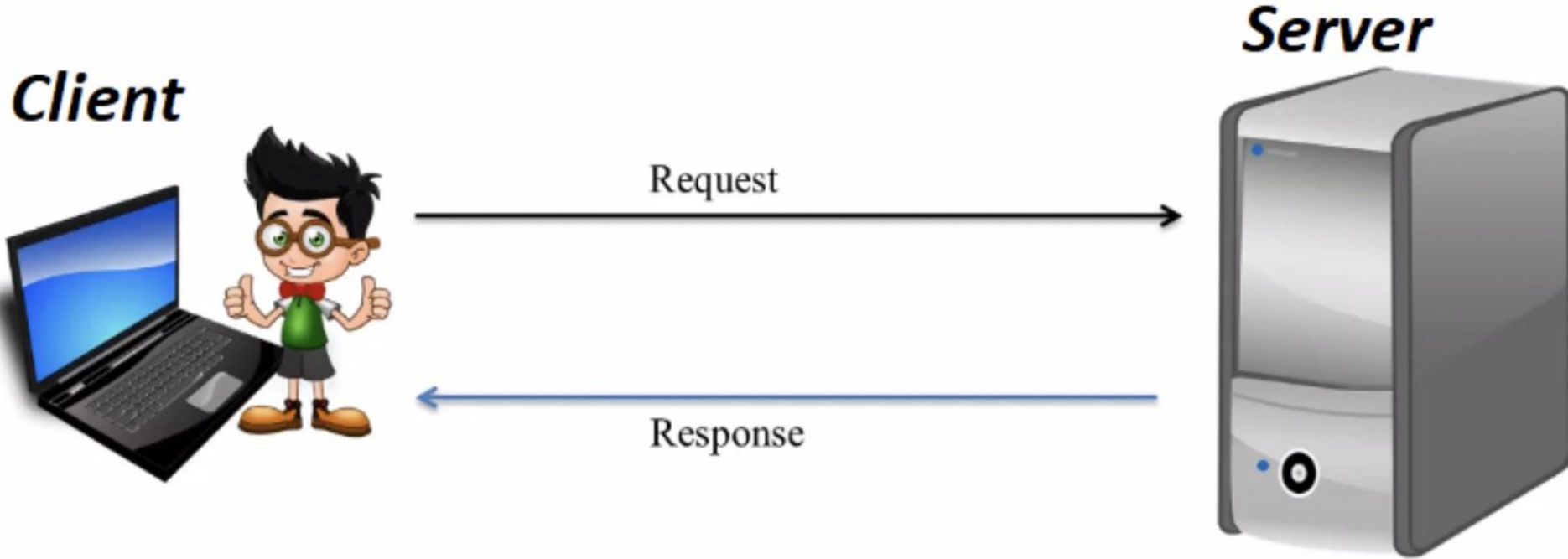
- Communication flow and Page loading in Browser
- What is DOM ?
- DOM Structure
-



Client Server Architecture

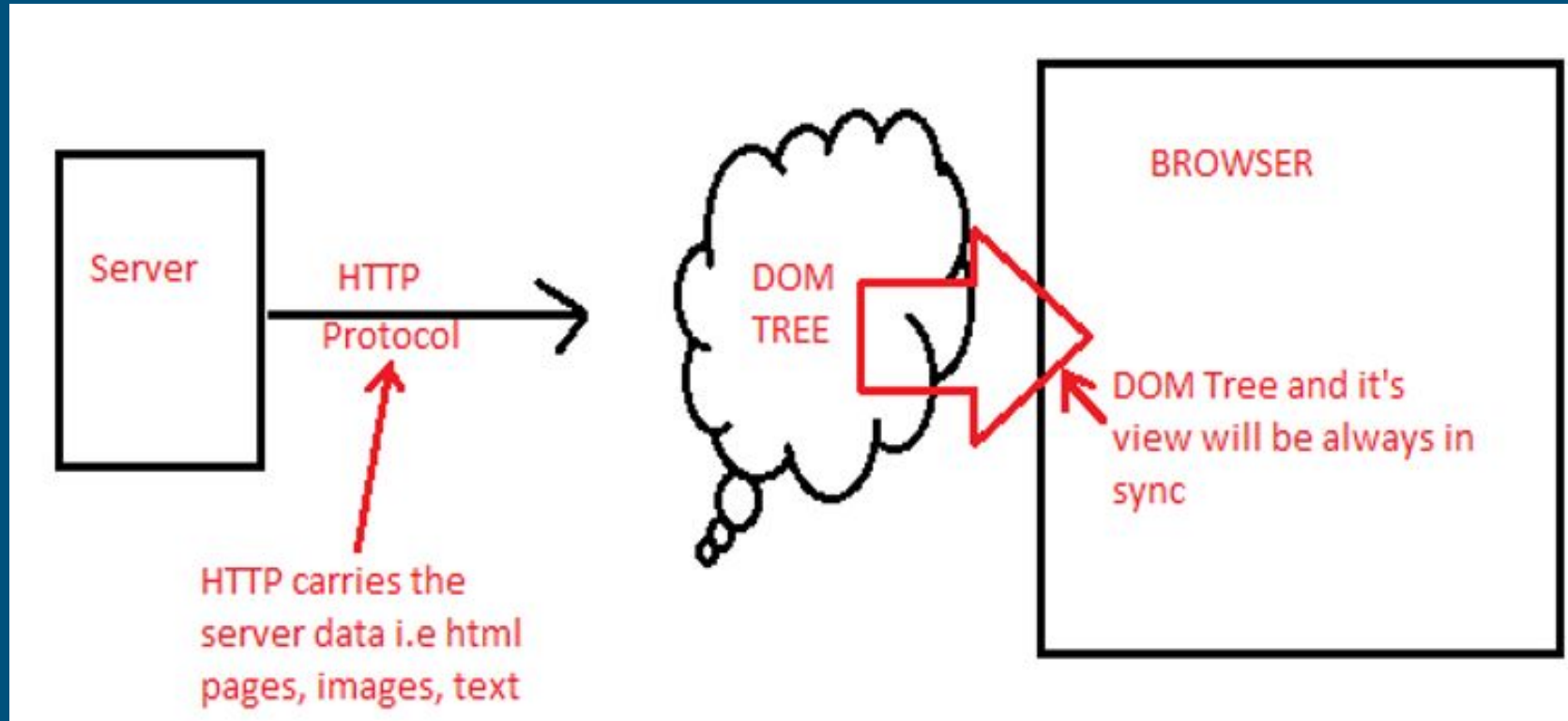
Client: A client is a piece of computer hardware or software that accesses a service which made available by Server. Ex. Browser

Server: A server is a Computer program or a device that provides functionality for other programs or a device called client.



DOM - Document Object Model

- The DOM defines a standard for accessing documents
- In the DOM all HTML elements are defined as objects

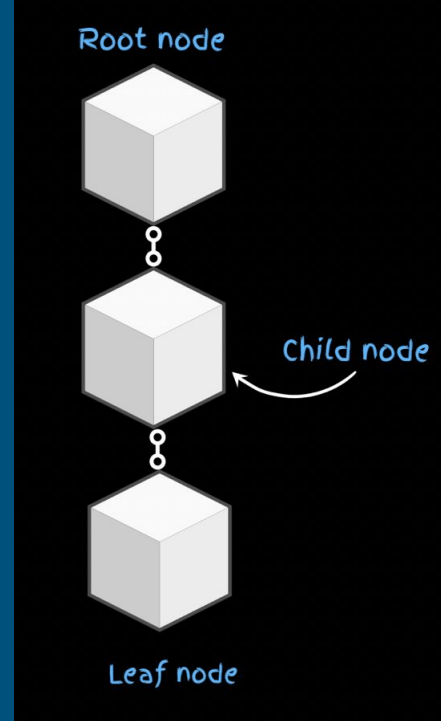


DOM - Document Object Model

- When a web page is loaded, the browser creates a Document Object Model of the page.
- The DOM defines a standard for accessing documents
- The Document Object Model represents each of these web pages in a tree-like structure to make it easier to access and manage the Elements.
- The DOM is the tree of nodes corresponding to HTML elements on a page

DOM Defines

- The HTML elements as objects
- The properties of all HTML elements
- The methods to access all HTML elements
- The events for all HTML elements



window object

- Window is the main container or global object. Any operation related to entire browser window can be a part of window object
- Window has methods, properties and objects. Ex. `setTimeout()` or `setInterval()`. Where as document is the object of the Window
- And It also has a screen object with properties describing the physical display.

If we want to see the location of our page that we are in → `window.location` or `window.location.href`
`console.dir(document)` → to see the complete document object

document

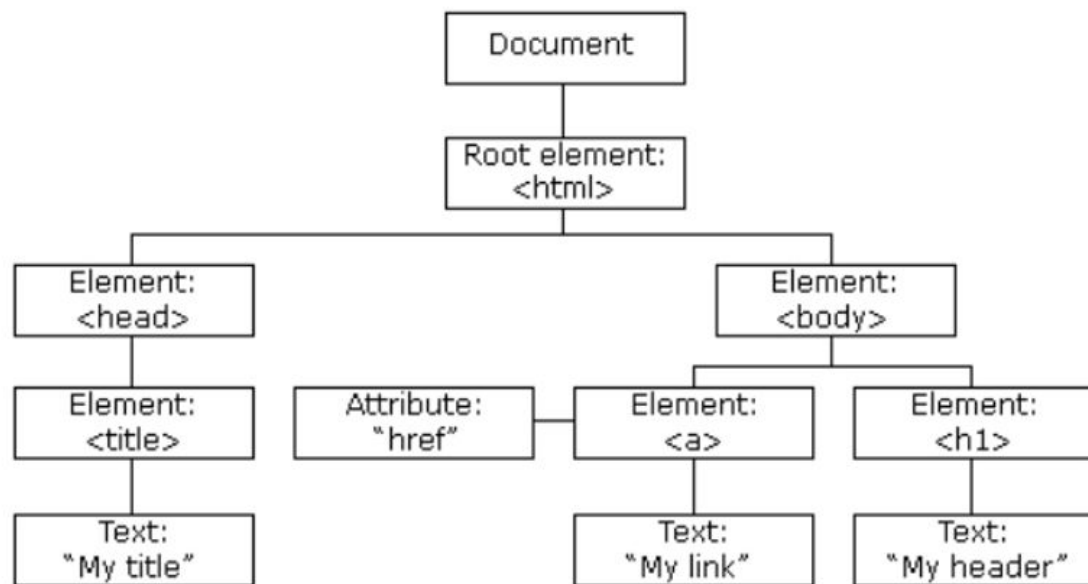
- The DOM is the child of window object
- Document is used to render the HTML elements
- Inspect in browser and type on console window, we can see document is the child object window

```
<!DOCTYPE html>
<html>
  <head>
    <title> My Title</title>
  </head>

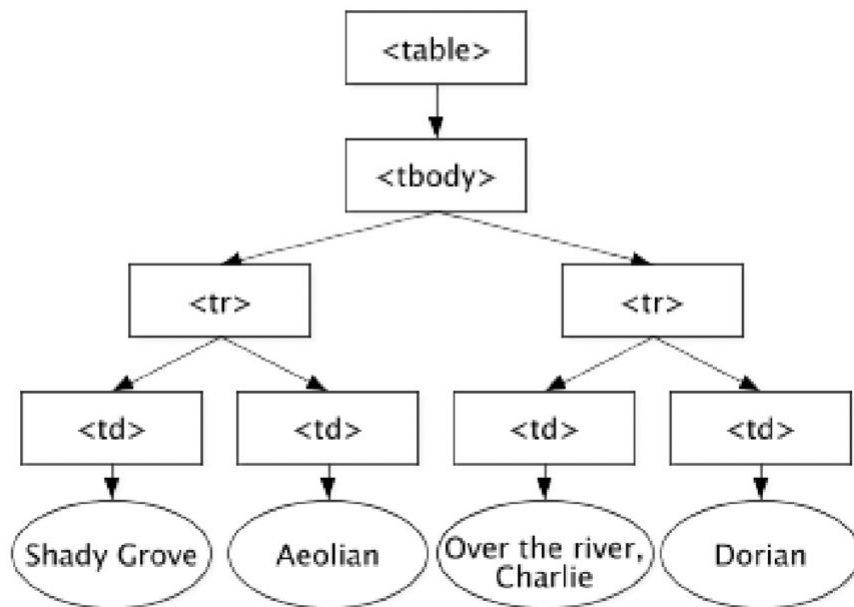
  <body>
    <a href="#"> My Link</a>
    <h1> My Header</h1>
  </body>

</html>
```

HTML code it's DOM tree representation



The Document Object Model of this code can be created like this:



```
<table>
  <tbody>
    <tr>
      <td>Shady Grove</td>
      <td>Aeolian</td>
    </tr>
    <tr>
      <td>Over the River, Charlie</td>
      <td>Dorian</td>
    </tr>
  </tbody>
</table>
```


With the object model, JavaScript gets all the power it needs to create dynamic HTML

- JavaScript can create new HTML events in the page
- JavaScript can change all the HTML elements in the page
- JavaScript can remove existing HTML elements and attributes
- JavaScript can add new HTML elements and attributes
- JavaScript can react to all existing HTML events in the page
- JavaScript can change all the CSS styles in the page
- JavaScript can change all the HTML attributes in the page

List of common APIs in web and XML page scripting using the DOM.

- `document.getElementById('element_id')`
- `document.querySelector(selector)`
- `document.querySelectorAll(name)`
- `document.createElement(name)`
- `parentNode.appendChild(node)`
- `element.innerHTML`
- `element.style.left`
- `element.setAttribute()`
- `element.getAttribute()`
- `element.addEventListener()`
- `window.content`

querySelector(), getElementById()

- Return single element
- Different ways of querying element (By CSS selector, by ID)
- Direct reference to DOM element is returned

querySelectorAll(), getElementsByTagName()

- Return collection of elements (array - like objects): NodeList
- Different ways of querying element (By CSS selector, by Tag Name, by CSS class)
- querySelectorAll() returns a non-live NodeList, getElementsByTagName() returns a live NodeList

What is a Node in the context of DOM?

Every item in the DOM tree is called a node, There are two types of node

1. **An Element Node:** Node that has an element
2. **Text Node:** Node that has text

Child Node: A node which is a child of another node

Parent Node: A node which has one or more child

Sibling Node: A node that share the same parent node

Descendant Node: A node which is nested deep in the tree

Finding HTML Element

`getElementById('element_id');`

`querySelector(selector), querySelectorAll(selector);`

JS script.js X

JS script.js > ...

```
1 console.log('Select an element using getElementById()');
2 const elementDetails = document.getElementById('details');
3 console.log(elementDetails);
4 console.log(elementDetails.innerHTML);
5
6 console.log('Select an element using querySelector()');
7 const elementDetail = document.querySelector('#details');
8 console.log(elementDetail);
9
10 console.log('Select an element using querySelector()');
11 const elementTechStack = document.querySelector('.techStack');
12 console.log(elementTechStack);
13
14 console.log('Select an element using querySelectorAll()');
15 const elementsTechStack = document.querySelectorAll('.techStack');
16 console.log(elementsTechStack[0]);
17 console.log(elementsTechStack[1]);
```

<> profile.html X

<> profile.html > html > body > script

```
1 <!DOCTYPE html>
2 <html>
3   <head><title> DOM - Document Object Model </title> </head>
4   <body>
5     <h2 id="details"> My Profile details</h2>
6     <h3 class="techStack"> Technology stack is </h3>
7     <ul class="techStack">
8       <li> HTML5 and CSS </li>
9       <li> Angular </li>
10      <li> React </li>
11      <li> Bootstrap </li>
12    </ul>
13    <script src="script.js"></script>
14  </body>
15</html>
```

Assignment to select DOM elements

Exercise: Selecting Elements in the DOM

Practice your DOM selection skills!

Your task is to select and store three DOM elements in the pre-defined variables

`mainHeading`, `secondAdvantage` and `advDiv`:

- `mainHeading` should store the `<h1>` element
- `secondAdvantage` should store the second (!) `` element (`Available`)
- `advDiv` should store the `<div>` with the id `advantages`

Use the DOM selection tools you learned about to get access to those elements and store them in the three variables.

Write your code inside script tag line no. 22, 23 and 24 as show here

```
1 <!DOCTYPE HTML>
2 <html>
3   <head>
4     <title>Exercise time!</title>
5   </head>
6   <body>
7     <main>
8       <div id="overview">
9         <h1>HTML & JavaScript are crucial technologies
10        </h1>
11        <p>The web would not work without them...</p>
12      </div>
13      <div id="advantages">
14        <ul>
15          <li>Flexible</li>
16          <li>Available</li>
17          <li>Magical</li>
18        </ul>
19      </div>
20    </main>
21    <script>
22      // Your solution code goes here
23      const mainHeading = ...
24      const secondAdvantage = ...
25      const advDiv = ...
26    </script>
27  </body>
28 </html>
```

HTML & JavaScript are crucial technologies

The web would not work without them...

- Flexible
- Available
- Magical

My Profile Details

Change the color of this element in JS to your favourite color

Technology Stack

- HTML and CSS
- Agile and JIRA
- Angular or React
- Oracle Database

Remove this child

My Hobbies

[My Profile](#)

Project Description

WAP to check the given string is palindrome or not

1. Function name → isPalindrome() with one argument that is str
2. Call this function for below strings
 - a. "madam"
 - b. "141"
 - c. "Sunday"
 - d. "mom"
 - e. "listen"
 - f. "dad"

Please log the output with meaningful message

Changing the text of an element

```
<h3 class="techStack"> Technology stack is </h3>
```



```
const elementTechStackH3 = document.querySelector('h3.techStack');  
elementTechStackH3.innerHTML = 'Complete Technology Stack is:';
```

Changing attribute of an element

```
<a href="https://www.google.com/" id="profileLink"> Social Media</a>
```



```
const elementProfileLink = document.querySelector('#profileLink');  
elementProfileLink.setAttribute('href', 'https://www.linkedin.com/');
```

Changing style property of an element

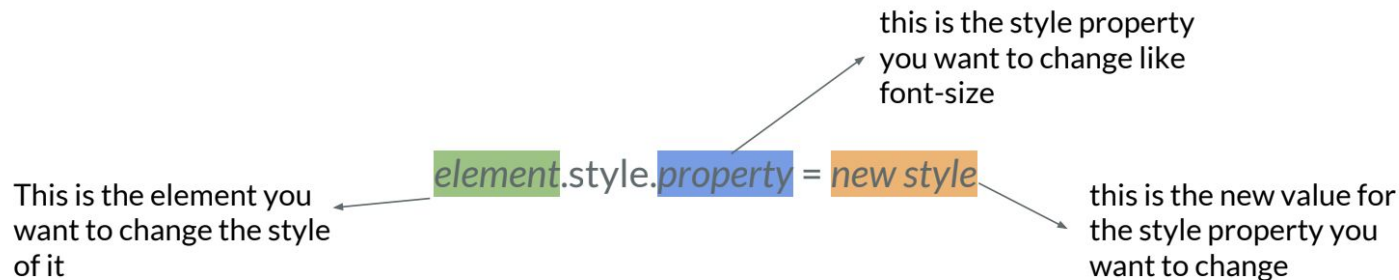
```
<h2 id="details"> My Profile details</h2>
```



```
const elementProfileDetails = document.getElementById('details');  
elementProfileDetails.style.color = 'blue';  
elementProfileDetails.style.fontFamily = 'Arial';  
elementProfileDetails.style.fontSize = 'larger';
```

Changing CSS properties

To change the style of an HTML element, use this syntax:



```
<h2 id="details"> My Profile details</h2>
```



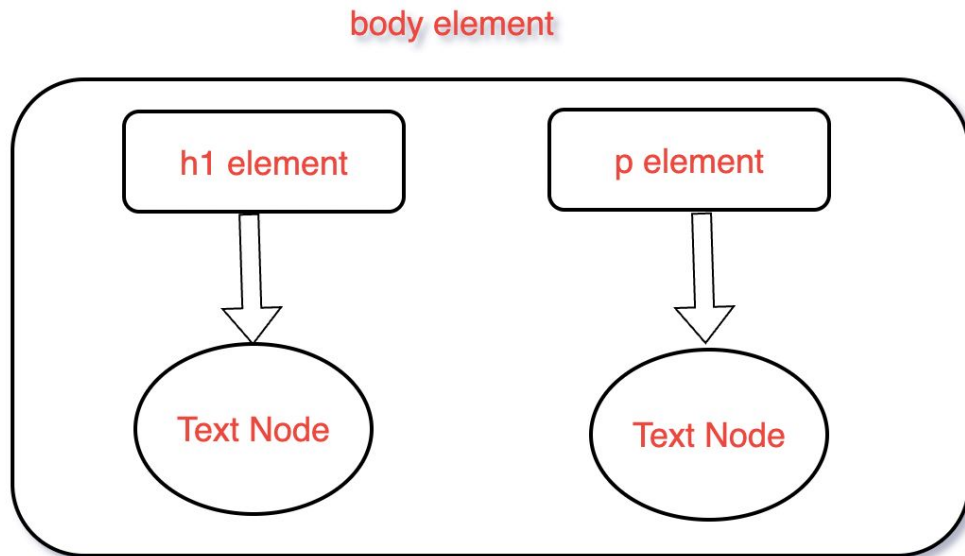
```
const elementProfileDetails = document.getElementById('details');
```

```
elementProfileDetails.style.color = 'blue';
```

```
elementProfileDetails.style.fontFamily = 'Arial';
```

```
elementProfileDetails.style.fontSize = 'larger';
```

How to create node:

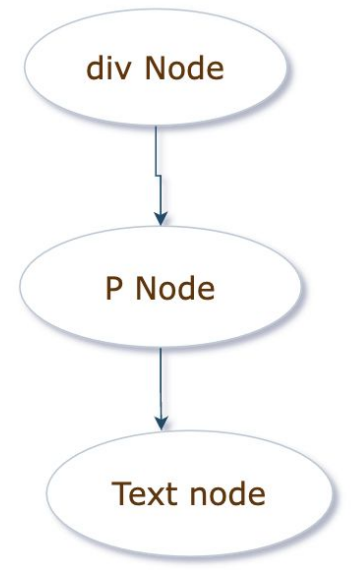


```
<script>
// run this function when the document is loaded
window.onload = () => {
  const heading = document.createElement("h1");
  const headingText = document.createTextNode("This is h1 element");
  heading.appendChild(headingText);
  document.body.appendChild(heading);

  const para = document.createElement("p");
  const paraText = document.createTextNode("This is paragraph");
  para.appendChild(paraText);
  document.body.appendChild(para);
};
</script>
```

Creating an element inside div tag

```
<div id="divSection">  
  <h3> This is div section </h3>  
</div>
```



```
const elementDiv = document.querySelector('#divSection');  
const elementPara = document.createElement('p'); // create p element  
elementPara.style.color = 'orange';  
elementPara.style.fontSize = '25px';  
const textElement=document.createTextNode("I am leaf node"); //leaf node  
elementPara.appendChild(textElement); // append lead node to parent node  
elementDiv.appendChild(elementPara); //append p element to it's parent div node
```

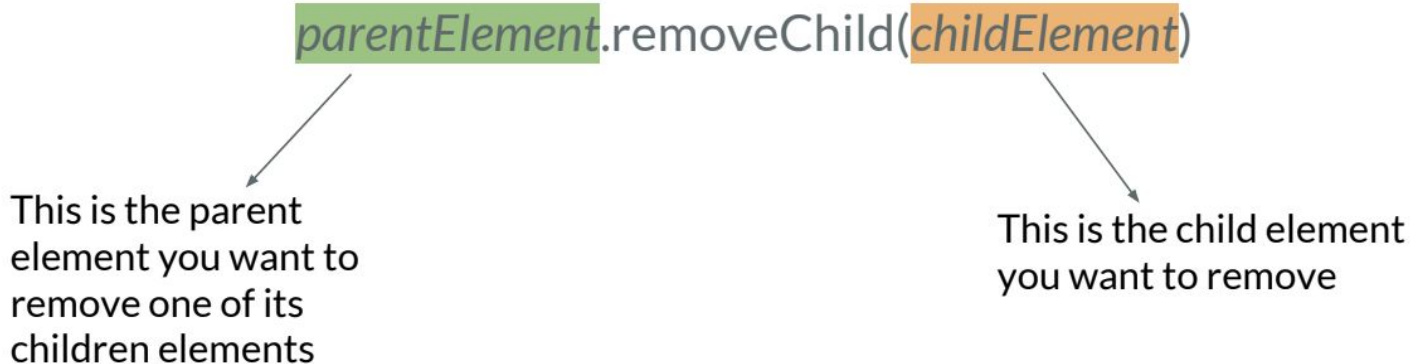
Removing Existing HTML Element

Removing Existing HTML Elements

To remove an HTML element, you must know the parent of the element

Then you can use this syntax to remove the element you want:

`parentElement.removeChild(childElement)`



This is the parent element you want to remove one of its children elements

The diagram illustrates the syntax of the `removeChild()` method. It shows `parentElement` in a green box and `childElement` in an orange box. An arrow points from `parentElement` to the explanatory text on the left, and another arrow points from `childElement` to the explanatory text on the right.

This is the child element you want to remove

Removing Existing HTML Element

Removing Existing HTML Elements

```
<div id="div1">  
  <p id="p1">This is a paragraph.</p>  
  <p id="p2">This is another paragraph.</p>  
</div>
```

```
let parent = document.querySelector("#div1");  
let child = document.querySelector("#p1");  
parent.removeChild(child);
```

DOM Events

Javascript DOM Events

Javascript can react to HTML DOM events

To achieve this we have to add an event listener that fires when a user causes any event e.g. clicks a button

`element.addEventListener(event, eventHandler)`

This is the element you want to capture the events on

This is the event you want to capture e.g. 'click' or 'mouseover'

This is the name of the function you want to call when the event is fired

__Event Types in JavaScript



Mouse Events

- click
- doubleclick
- mouseover
- mouseout
- mousemove

Form Events

- input
- submit
- change

Keyboard Events

- input
- keydown
- keypress
- keyup

User Interface Events

- Load
- Unload
- Error
- Resize
- Scroll

Focus and Blur Event

- focus
- blur
- focusin
- focusout

Handling 'click' event

```
<h3 onclick="show()"> This is to show event handle </h3>
```

1st way: Handling event using normal function

```
function show(){  
  console.log("Hey you clicked me!");  
}
```

2nd Way: Handling click event

Using addEventListener() →

```
const elementEventH3 = document.getElementById('h3Event');  
elementEventH3.addEventListener('click', () => {  
  console.log('Hello you clicked me')  
});
```

3rd way: Handling Click event using addEventListener()

```
const elementEventH3 = document.getElementById('h3Event');  
elementEventH3.addEventListener('click', function() {  
  console.log('Hello you clicked me');  
});
```

'mouseover' and 'mouseout' event

Change the button
Background color
on mouseover and mouseout
event

```
<h3 id="h3Event"> This is to show event handle </h3>
```

```
elementEventH3.addEventListener('mouseover', ()=>  
|   console.log("Hey you overed me ");  
});
```

```
<button id="btnSubmit">Submit</button>
```

```
const btnSubmit=document.getElementById('btnSubmit');  
btnSubmit.addEventListener('mouseover', () => {  
|   btnSubmit.style.background = 'cadetblue';  
});
```

```
btnSubmit.addEventListener('mouseout', () => {  
|   btnSubmit.style.background = 'orange';  
|   btnSubmit.style.borderRadius = '10px';  
});
```