DOM Power -7(30 coding exercises)

"Your calling should be a pointer for your skill development" — S. Adeleja

7.1 What is the DOM?

Exercises

1. Can you explain the DOM?
2. What are the 2 purposes of the DOM?
3. The DOM API can only be used with JavaScript. True or False?
4. How are objects in the DOM assembled? Choose one option:
    - Queue
    - Hierarchy
    - Stack

Answers

1. The DOM API allows you to access and manipulate HTML elements on webpages. You can add, delete and change HTML elements via the DOM API using JavaScript.
2. The two purposes of the DOM API are:
    - It defines the logical structure of an HTML document for example index.html
    - Dictates how the elements within an HTML document can be accessed and modified
3. False. The DOM API is language neutral programming API.
4. Hierarchy. Objects in the DOM are constructed in a hierarchical tree of objects.

7.2 Properties of the DOM

Exercises:

1. What root DOM node provides access to all the other DOM nodes?
2. Access the text content in between the opening and closing `<p>` tags:

```
<p id='myP'>The DOM is super handy for making webpages
interactive</p>
```

3. Change the text content in between the `<h1>`tags to `Enterprise Solutions`

```
  <h1>WebTech Solutions<h1>
```

4. Map the following input attributes to DOM properties

```
<input type='number' id='productVal' value= 100
status='available'>
```

5. Return the children of the following `<ul>` element

```
<ul>
     <li>ID: 2183</li>
     <li>ID: 1030</li>
     <li>ID: 105</li>
     <li>ID: 8294</li>
</ul>
```

6. Change the color of the text within the first `<p>` tag to `#b19cd9`

```
<p>id quod maxime placeat facere possimus, omnis
volupt</p>

  <p>Sed ut perspiciatis unde omnis iste natus error sit
voluptatem accusantium doloremque laudantium, totam rem
aperiam, eaque ipsa quae ab illo inventore veritatis et
quasi architecto beatae vitae dicta sunt explicabo. </p>

  <p>Ut enim ad minima veniam, quis nostrum
exercitationem ullam corporis suscipit laboriosam, nisi
ut aliquid ex ea commodi consequatur?</p>
```

7. Change the color, background color and width properties of the following `div` element. The color should be #77dd7, background color should be #fdfd96 and the width should be set to 100px.

```
<div id='div1'>

   <p>Arrays rock</p>

</div>
```

Answers

1. `document` is the root object/node which will allow you to access all the other DOM nodes.

2. We can verify that we've accessed the text content of the `myP` element by retrieving the element and then accessing the value of its `innerHTML` property. This is assigned to a variable `paraText` and we can `console.log` `paraText` to verify its value

```
let paraText = document.getElementById('myP').innerHTML;

console.log(paraText);
```

```
"The DOM is super handy for making webpages interactive"
```

3.

```
let heading1 = document.querySelector('h1').innerHTML =
'Enterprise Solutions';
```

4. Only the standard HTML attributes are converted to DOM properties.

```
console.log(document.getElementById('productVal').type);
// "number"

console.log(document.getElementById('productVal').id);
//"productVal"

console.log(document.getElementById('productVal').value);
// "100"

console.log(document.getElementById('productVal').status)
; // undefined
```

5.

```
let children =
document.querySelector('ul').getElementsByTagName('li');
```

```
console.log(children.length); //4
```

6. The `querySelector()` method returns the 1st instance of a query:

```
let firstP = document.querySelector('p');

firstP.style.color = "#b19cd9";
```

7. Retrieve the `div` element and use the `style` property to modify the `div`:

```
let div = document.getElementById('div1');

div.style.color = '#77dd77';

div.style.width = '100px';

div.style.backgroundColor = '#fdfd96'
```

7.3 DOM methods

Exercises

1. What are 3 DOM methods used to retrieve DOM nodes/HTML elements
2. What is the difference between `querySelector()` and `querySelectorAll()` ?
3. For the following HTML, retrieve the number of `<p>` elements

   Hint: use the `length` *property*:

```
<ul>
 <p>foo</p>
 <p>bar</p>
 <p>zinga</p>
 <p>nee</p>
</ul>
```

4. For the above HTML (#3) retrieve the `innerHTML` of the fourth `<p>`
5. Carrying on from question #3 and #4, remove the third `<p>`
6. Create a `<div>` and `<p>`. Assign the following text to the `<p>` and append it to the `<div>`. Make sure that both elements are present on a webpage

`'Cras ultrices, risus vitae tristique malesuada, metus tortor pulvinar eros, id finibus ex neque id magna. In lectus tortor, sagittis et purus in, vulputate viverra massa. Nullam in erat diam. Sed eleifend, eros ac venenatis convalli'`

7. From question #6, replace the `<p>` node with a `<li>` node. Set the text content of the `<li>` to `'Magnus Operandi'` Don't delete anything.

8. For the following HTML insert the number 100 before 200:

```
<ul id='myUl'>
  <li>200</li>
  <li>300</li>
  <li>400</li>
</ul>
```

9. `innerHTML` is preferred over the `textContent` property? True or False?

10. `console.log` the value of the third input element (`'Pinky'`) using the `querySelectorAll()` method.

```
<label for='username'>Username:</label>
<input type='text' id='username' name='username'value=''>
<button id='myButton'>Submit</button>


<label for='username1'>Username:</label>
<input type='text' id='username1'
name='username1'value='Gigi'>
<button id='myButton'>Submit</button>


 <label for='username2'>Username:</label>
<input type='text' id='username2'
name='username2'value='Pinky'>
<button id='myButton'>Submit</button>
```

11. Create a button and insert it into your document. Set the text content of the button to 'click'
12. Carrying on from question #11, set the `id` attribute of the button to 'myButton' and set its `onclick` attribute to 'clickFunc()'. Log the button to the console, what do you get back?
13. Attach an `addEventListener()` method to the button so that when the button is clicked, the user is alerted with the greeting, 'Hi there junior dev'
14. Get the onclick attributes of the button that you just created with the `getAttribute()` method
15. What is the difference between the document and window object?
16. What is the difference between `remove()` and `removeChild()`?

Answers

1. 3 DOM methods used to retrieve DOM nodes/HTML elements are:
    - getElementById()
    - querySelector()
    - getElementsByTagName
2. The `querySelector()` methods returns the first query. Whereas the `querySelectorAll()` method will return all the queried nodes
3.

```
let pFourth = document.querySelectorAll('p');
console.log(pFourth.length); // 4
```

4.

```
let pEl = document.querySelectorAll('p');
console.log(pEl[3].innerHTML); // "nee"
```

5.

```
let pEl = document.querySelectorAll('p');
let ul = document.querySelector('ul');
```

```
ul.removeChild(pEl[2]);
```

6. .

```
let div = document.createElement("div");

let p = document.createElement("p");

p.textContent = 'Cras ultrices, risus vitae tristique
malesuada, metus tortor pulvinar eros, id finibus ex
neque id magna. In lectus tortor, sagittis et purus in,
vulputate viverra massa. Nullam in erat diam. Sed
eleifend, eros ac venenatis convallim';

div.appendChild(p);

document.body.appendChild(div);
```

7.

```
let div = document.createElement("div");

let p = document.createElement("p");

p.textContent = 'Cras ultrices, risus vitae tristique
malesuada, metus tortor pulvinar eros, id finibus ex
neque id magna. In lectus tortor, sagittis et purus in,
vulputate viverra massa. Nullam in erat diam. Sed
eleifend, eros ac venenatis convallim';

div.appendChild(p);

let li = document.createElement('li');

li.textContent = 'Magnus Operandi';

div.replaceChild(li, div.children[0]);

document.body.appendChild(div);
```

8.

```
let ul = document.getElementById('myUl');

let li = document.createElement('li');

li.textContent = '100';

ul.insertBefore(li, ul.firstElementChild);
```

9. False. `textContent` is preferred as `innerHTML` is at risk of XSS attacks.

10. The `querySelectorAll()` method returns a node list therefore, you must specify the index of the input element you want:

```
let input = document.querySelectorAll('input');
console.log(input[2].value);
```

11.

```
let button = document.createElement('button');
button.textContent = 'click';
document.body.appendChild(button);
```

12.

```
let button = document.createElement('button');
button.textContent = 'click';
button.setAttribute('id', 'myButton');
button.setAttribute('onclick', 'myFunc()');
document.body.appendChild(button);
console.log(button);
```

```
"<button id='myButton' onclick='myFunc()'>click</button>"
```

13.

```
let button = document.createElement('button');
button.textContent = 'click';
button.setAttribute('id', 'myButton');
button.setAttribute('onclick', 'myFunc()');
button.addEventListener('click', function(){
  alert('Hi there junior dev')
})
```

14.

```
button.getAttribute('onclick')
```

15. The global window object is loaded into your browser, it has methods and properties such as `window.onload()` etc. The document is a document containing your code which will be loaded inside the window.
16. `remove()` will remove the entire parent element. `removeChild()` will remove the first child node of a specified parent.

7.4 Iterating through DOM nodes

Exercises

1. What is **NodeList**?
2. Iterate through the following `<p>` elements and append the string `'Bonus Player'` to each one

```
<div class='container'>
<div class='row'>
   <div class='col-sm-12'>
      <p>BaaBara</p>
      <p>Poco</p>
      <p>Stardew</p>
      <p>LikeK</p>
   </div>
</div>
<div>
```

3. Remove the 2nd `<p>` from the div

Answers

1. A **NodeList** is a collection of nodes retrieved from a document. It is an array-like object therefore, you can iterate through it. However, it is not an array and array methods cannot be used.
2.

```
let pEl = document.querySelectorAll('p');
pEl.forEach(function(item){
```

```
  item.textContent += ' Bonus Player';

})
```

3.

```
let div = document.getElementsByClassName('col-sm-
12')[0];

div.removeChild(pEl[1]);
```