DOM (Document Object Model)

The DOM (Document Object Model) is a programming interface for web documents. It represents the page structure as a logical tree, allowing JavaScript to access and manipulate the content, structure, and style of a document. Essentially, it's how JavaScript "sees" and interacts with an HTML or XML page.

DOM Tree Structure

When a web page is loaded, the browser creates a DOM tree, which is a hierarchical representation of the HTML document. Each HTML element, attribute, and text content becomes a "node" in this tree.

Here's a simple HTML example and its corresponding DOM tree structure:

HTML:

```html
<!DOCTYPE html>
<html>
<head>
  <title>My Page</title>
</head>
<body>
  <h1 id="main-title" class="header">Welcome</h1>
  <p>This is a <span>paragraph</span>.</p>
  <ul>
    <li>Item 1</li>
    <li>Item 2</li>
  </ul>
</body>
</html>
```

DOM Tree Representation:

```
Document
└── html
    ├── head
    │   └── title
    │       └── "My Page" (Text Node)
    └── body
        ├── h1 (id="main-title", class="header")
        │   └── "Welcome" (Text Node)
        ├── p
        │   ├── "This is a " (Text Node)
        │   └── span
        │       └── "paragraph" (Text Node)
        ├── ul
            ├── li
            │   └── "Item 1" (Text Node)
            └── li
               └── "Item 2" (Text Node)
```

In this tree:

- **Document** is the root node.
- **Elements** (like html, head, body, h1, p, span, ul, li, title) are element nodes.
- **Attributes** (like id="main-title", class="header") are attribute nodes (though often conceptualized as properties of the element node).
- **Text** inside elements (like "My Page", "Welcome") are text nodes.

DOM Selectors

DOM selectors are methods used in JavaScript to find and select HTML elements within the DOM tree. Once an element is selected, you can manipulate its properties, content, or style.

Here are the most common DOM selectors with examples:

1. **document.getElementById()**:
    - Selects a single element by its unique id attribute.
    - Returns the element object or null if not found.

Example:

```
 <h1 id="myHeading">Hello DOM</h1>
<script>
    const heading = document.getElementById('myHeading');
    console.log(heading.textContent); // Output: Hello DOM
    heading.style.color = 'blue';
</script>
```

2. **document.getElementsByClassName()**:
    - Selects all elements that have a specific class name.
    - Returns an HTMLCollection (live collection, similar to an array) of elements.

   Example:

```
<p class="my-text">Paragraph 1</p>
<p class="my-text">Paragraph 2</p>
<script>
    const paragraphs = document.getElementsByClassName('my-text');
    for (let i = 0; i < paragraphs.length; i++) {
        console.log(paragraphs[i].textContent);
        // Output: Paragraph 1, Paragraph 2
    }
    paragraphs[0].style.fontWeight = 'bold';
</script>
```

3. **document.getElementsByTagName()**:
    - Selects all elements of a specified tag name (e.g., div, p, a).
    - Returns an HTMLCollection.

   Example:

```
<ul>
    <li>Apple</li>
    <li>Banana</li>
</ul>
<script>
    const listItems = document.getElementsByTagName('li');
    console.log(listItems.length); // Output: 2
    listItems[0].style.backgroundColor = 'yellow';
```

```
</script>
```

4. **document.querySelector()**:
    - Selects the *first* element that matches a specified CSS selector.
    - Returns the element object or null if not found.

    Example

```
<div class="container">
   <p>First paragraph</p>
   <p class="highlight">Second paragraph</p>
</div>
<script>
   const firstParagraph = document.querySelector('.container p');
   console.log(firstParagraph.textContent); // Output: First paragraph

   const highlightedParagraph = document.querySelector('.highlight');
   console.log(highlightedParagraph.textContent); // Output: Second paragraph
</script>
```

5. **document.querySelectorAll()**:
    - Selects *all* elements that match a specified CSS selector.
    - Returns a NodeList (non-live collection, similar to an array).

Example:

```
<button class="btn">Click Me 1</button>
<button class="btn">Click Me 2</button>
<script>
   const buttons = document.querySelectorAll('.btn');
   buttons.forEach(button => {
      button.addEventListener('click', () => {
         alert(`Button ${button.textContent} clicked!`);
      });
   });
</script>
```

Choosing the right selector:

- Use getElementById when you need to select a unique element by its ID.
- Use getElementsByClassName or getElementsByTagName when you need to select multiple elements based on their class or tag.
- Use querySelector when you need to select the first element matching a complex CSS selector.
- Use querySelectorAll when you need to select all elements matching a complex CSS selector.