DOM Content Manipulation

Understanding the concept of JavaScript properties, such as `**.innerHTML**`, and `**.textContent**` is vital for effectively manipulating and accessing content within HTML elements.

.innerHTML:

- Usage: Used to get or set the HTML content within an element.
- Returns: A string representing the HTML content, including tags and markup.
- Effect: Can be used to dynamically update or retrieve the markup and text within an element.
- Example:

```
<!DOCTYPE html>
<html>
  <head>
    <title>Example</title>
  </head>
  <body>
    <div id="myElement">Initial content</div>


    <script>
      var element = document.getElementById("myElement");
      var htmlContent = element.innerHTML;
      console.log( htmlContent);
      // Output:Initial content
```

```
      element.innerHTML = "<strong>New content</strong>";

      var updatedHtmlContent = element.innerHTML;

      console.log("Updated HTML content:", updatedHtmlContent);

      // Output:  <strong>New content</strong>

    </script>

  </body>

</html>
```

.textContent:

- Usage: Used to get or set the text content of an element, excluding any HTML tags or markup.
- Returns: A string representing the plain text content without any HTML formatting.
- Effect: Allows manipulation of the text content specifically.
- Example:

```
<!DOCTYPE html>
<html>
  <head>
    <title>Example</title>
  </head>
  <body>
    <div id="myElement">Initial text content</div>

    <script>
      var element = document.getElementById("myElement");
```

```
    var textContent = element.textContent;

    console.log("Initial text content:", textContent);

    // Output: Initial text content: Initial text content


    element.textContent = "New text content";

    var updatedTextContent = element.textContent;

    console.log("Updated text content:", updatedTextContent);

    // Output: Updated text content: New text content

  </script>

 </body>

</html>
```

- **Note:** **`.textContent`** provides a safe way to modify the textual content without the risk of executing potential malicious code.

.innerHTML vs .textContent

a. **`.innerHTML`** is primarily used when you need to work with the HTML content of an element, including tags and markup. It allows you to retrieve or modify both the text and HTML structure within an element.

b. This property is useful when you want to update or extract HTML content, such as inserting new elements, modifying attributes, or applying formatting.

c. For instance, if you need to dynamically insert new elements or modify the existing structure, .innerHTML is the appropriate choice.

d. On the other hand, **`.textContent`** is specifically designed to work with the text content of an element, excluding any HTML tags or markup.

e. It provides a straightforward way to manipulate or extract the plain text within an element, regardless of any HTML formatting. Use .textContent when you need to perform operations solely on the text, such as extracting data or updating the textual content without affecting the HTML structure or any child elements within the element.

f. Unlike **`.innerHTML`, `.textContent`** treats the assigned value purely as text, ensuring that it is not interpreted as HTML or executed as code, making it a safer option.