NAZARBAYEV UNIVERSITY

SCHOOL OF ENGINEERING AND DIGITAL SCIENCES

Web Programming and Problem Solving

DOM Manipulation

Date: 09.11.2022

Instructor: Zhandos Yessenbayev

NAZARBAYEV UNIVERSITY

Content

SCHOOL OF ENGINEERING AND DIGITAL SCIENCES

- **Introduction**
- **Finding Elements**
- **Element Manipulation**
- **Content Manipulation**
- **Attribute Manipulation**
- **Style Manipulation**
- **Class Manipulation**

What is DOM?

The HTML DOM is a standard **object model** and **programming interface** for HTML document.

DOM defines:

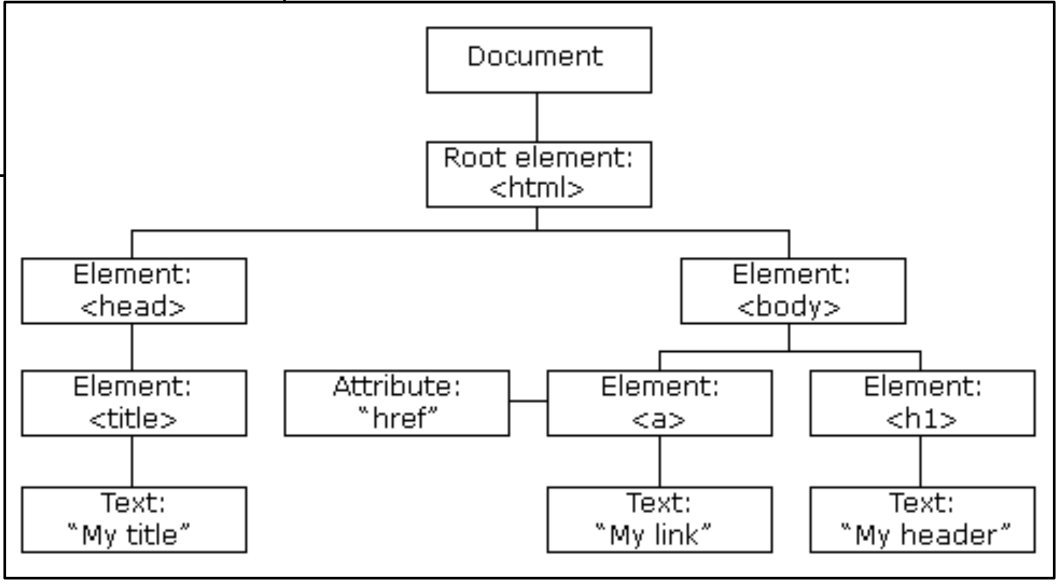
- The HTML elements as objects
- The properties of all HTML elements
- The methods to access all HTML elements
- The events for all HTML elements

What is DOM?

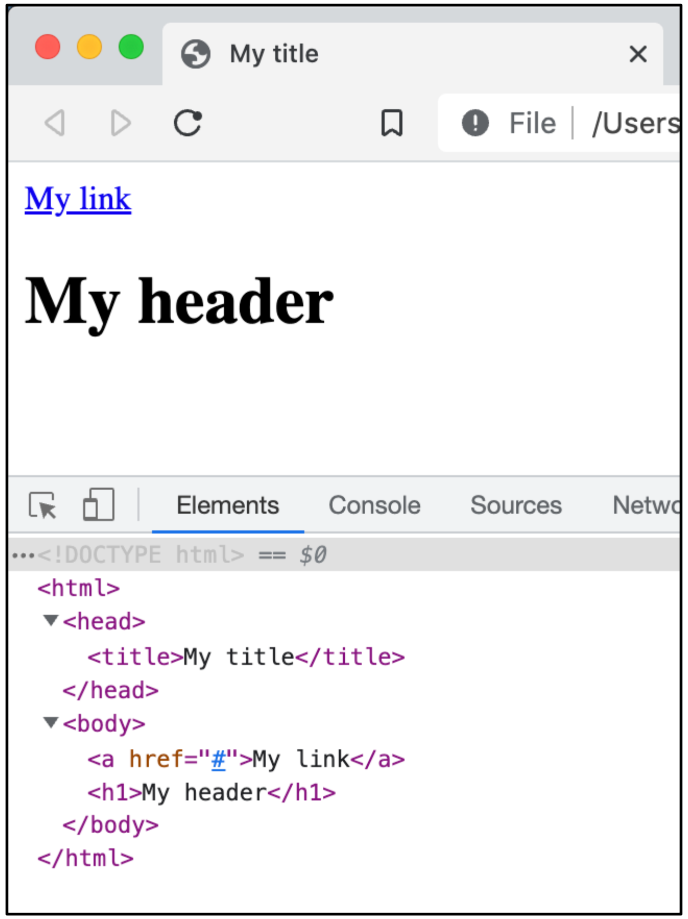
```
<!DOCTYPE html>
<html>
  <head>
    <title>My title</title>
  </head>
  <body>
    <a href="#">My link</a>
    <h1>My header</h1>
  </body>
</html>
```

HTML View

Graphical View



Browser View



My link

My header

```
…<!DOCTYPE html> == $0
<html>
▼<head>
    <title>My title</title>
  </head>
▼<body>
    <a href="#">My link</a>
    <h1>My header</h1>
  </body>
</html>
```

Document Object

The main object in the DOM is document. All the elements are accessible via document.

Main methods to access elements:

- Finding elements
- Creating elements
- Adding elements
- Deleting elements

Finding elements

The elements in DOM can be found by element's ID, tag and class names, CSS selector:

- document.getElementById(<element_id>)
- document.getElementsByTagName(<tag_name>)
- document.getElementsByClassName(<class_name>)
- document.querySelectorAll(<CSS selector>);

Note that the last three methods return an array of elements

Element Manipulation

The main methods to work with elements are:

- document.createElement(<element>)
- document.removeChild(<element>)
- document.appendChild(<element>)
- document.replaceChild(<new_element>, <old_element>)

NAZARBAYEV UNIVERSITY

Element Manipulation

- Example: Adding a header element to the body

```
// print the initial document object
console.log(document);

// create an HTML element – h1
let h1 = document.createElement("h1");

// append the h1-element to the body-element
document.body.appendChild(h1);

// print the final document object
console.log(document);
```

Note that the header element has no text, i.e. it is empty

Content Manipulation

- There are two basic properties to manipulate content of the elements (add, clear):
  - innerText
  - innerHTML

NOTE:
With innerHTML, the content is treated as an HTML content, i.e.properly decoding HTML tags.

With innerText, the content is treated as pure text.

```
// print the initial document object
console.log(document);

// create an HTML element - h1
let h1 = document.createElement("h1");

// create an HTML element - h1
h1.innerText = ”<i>Header Text</i>"

// append the h1-element to the body-element
document.body.appendChild(h1);

// print the final document object
console.log(document);
```

Attribute Manipulation

The attributes of an HTML element can be accessed and manipulated using:

- getAttribute(<attribute_name>)
- setAttribute(<attribute_name>, <attribute_value>)

```
// create an HTML element - a
let a1 = document.createElement("a");
a1.setAttribute("href", "https://w3schools.com");
a1.innerText = "W3Schools";
document.body.appendChild(a1);
```

Note: getAttribute() method returns null if there is no requested attribute

NAZARBAYEV UNIVERSITY

Style Manipulation

To change CSS style of an element, **style** property (attribute) can be used.

Note: CSS property with **dash** are converted to **camel** case:

- Ex: background-color -> backgroundColor

```
let p1 = document.createElement("p");
p1.innerText = "This is the first paragraph!";
p1.style.color = "red";
p1.style.backgroundColor = "yellow";
document.body.appendChild(p1);
```

Class Manipulation

The elements might define several classes which can be accessed using **classList** property

- classList returns a list of all the classes
- classList itself has methods to add(), remove() and toggle() a class

```
// add and remove a class to the element
let btn1 = document.getElementById("btn1");
console.log(btn1.classList);
btn1.classList.add("btn");
console.log(btn1.classList);
btn1.classList.remove("btn");
console.log(btn1.classList);
```

Summary

- **Key takeaways**:
  - **DOM** is a standard way to work with HTML document
  - **document** object is used to access other elements
  - With JavaScript all the elements, their content, attributes, styles and classes can be accesses and manipulated.

Thanks for Attention!