HAZZA INSTITUTE OF TECHNOLOGY

iqracity

MERN – ES6 + React

Module 09: DOM Model

Outline

Module 09

HAZZA INSTITUTE OF TECHNOLOGY

- DOM Model Understanding
- Understanding of methods document.getElementById()
- Understanding of methods document.getElementByClass Name()
- Understanding of methods document.getElementsByTag Name()
- Understanding of methods document.querySelector()
- Understanding of methods document.querySelectorAll()

JavaScript HTML DOM

HAZZA INSTITUTE OF TECHNOLOGY

*With the HTML DOM, JavaScript can **access** and **change** all the elements of an HTML document.*

When a web page is loaded, the browser creates a **D**ocument **O**bject **M**odel of the page.

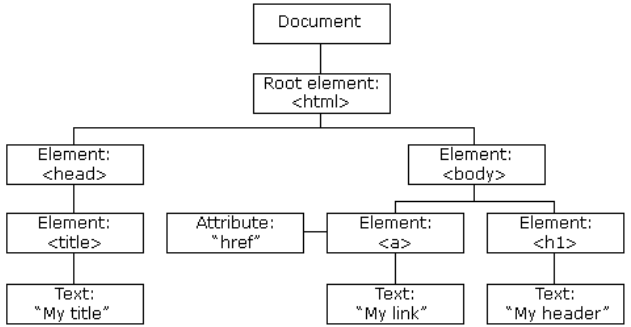
The HTML DOM is a standard **object** model and **programming interface** for HTML. It defines:

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

In other words: **The HTML DOM is a standard for how to get, change, add, or delete HTML elements.**

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

- JavaScript can change all the HTML elements in the page
- JavaScript can change all the HTML attributes in the page
- JavaScript can change all the CSS styles 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 create new HTML events in the page



The DOM Programming Interface

HAZZA INSTITUTE OF TECHNOLOGY

JavaScript HTML DOM

```
<html>
<body>

<p id="demo"></p>

<script>
document.getElementById("demo").innerHTML = "Hello World!";
</script>

</body>
</html>
```

The getElementById Method

The most common way to access an HTML element is to use the `id` of the element.

In the example above the `getElementById` method used `id="demo"` to find the element.

The innerHTML Property

The easiest way to get the content of an element is by using the `innerHTML` property.

The `innerHTML` property is useful for getting or replacing the content of HTML elements.

The DOM Programming Interface

HAZZA INSTITUTE OF TECHNOLOGY

*The HTML DOM document object is the **owner** of all other objects in your web page.*

Finding HTML Elements

Method	Description
document.getElementById(*id*)	Find an element by element id
document.getElementsByTagName(*name*)	Find elements by tag name
document.getElementsByClassName(*name*)	Find elements by class name

Changing HTML Elements

Property	Description
element.innerHTML = *new html content*	Change the inner HTML of an element
element.attribute = new value	Change the attribute value of an HTML element
element.style.*property = new style*	Change the style of an HTML element
Method	**Description**
element.setAttribute(*attribute, value*)	Change the attribute value of an HTML element

The DOM Programming Interface

HAZZA INSTITUTE OF TECHNOLOGY

*The HTML DOM document object is the **owner** of all other objects in your web page.*

Adding and Deleting Elements

Method	Description
document.createElement(*element*)	Create an HTML element
document.removeChild(*element*)	Remove an HTML element
document.appendChild(*element*)	Add an HTML element
document.replaceChild(*new, old*)	Replace an HTML element
document.write(*text*)	Write into the HTML output stream

Adding Events Handlers

Method	Description
document.getElementById(*id*).onclick = function() {*code*}	Adding event handler code to an onclick event

JavaScript Form Validation

HAZZA INSTITUTE OF TECHNOLOGY

JavaScript HTML DOM

Data Validation

Data validation is the process of ensuring that user input is clean, correct, and useful.

Typical validation tasks are:

- has the user filled in all required fields?
- has the user entered a valid date?
- has the user entered text in a numeric field?

Most often, the purpose of data validation is to ensure correct user input.

Validation can be defined by many different methods, and deployed in many different ways.

Server side validation is performed by a web server, after input has been sent to the server.

Client side validation is performed by a web browser, before input is sent to a web server.

Example 1

Example 2

Example 3

Changing HTML Style

JavaScript HTML DOM

HAZZA INSTITUTE OF TECHNOLOGY

- **Changing HTML Style**
  - The HTML DOM allows JavaScript to change the style of HTML elements.

```
document.getElementById(id).style.property = new style
```

Example 1

Example 2

Example 3

JavaScript Events

HAZZA INSTITUTE OF TECHNOLOGY

JavaScript basics

HTML Events

An HTML event can be something the browser does, or something a user does.

Here are some examples of HTML events:

- An HTML web page has finished loading
- An HTML input field was changed
- An HTML button was clicked

Often, when events happen, you may want to do something.

JavaScript lets you execute code when events are detected.

HTML allows event handler attributes, **with JavaScript code**, to be added to HTML elements.

```
<element event='some JavaScript'>
<element event="some JavaScript">
```

Example

JavaScript Events

JavaScript basics

HAZZA INSTITUTE OF TECHNOLOGY

Common HTML Events

Here is a list of some common HTML events:

Event	Description
onchange	An HTML element has been changed
onclick	The user clicks an HTML element
onmouseover	The user moves the mouse over an HTML element
onmouseout	The user moves the mouse away from an HTML element
onkeydown	The user pushes a keyboard key
onload	The browser has finished loading the page

Example 1

Example 2

Example 3

Web APIs

What is Web API?

HAZZA INSTITUTE OF TECHNOLOGY

A Web API is a developer's dream.

- API stands for **A**pplication **P**rogramming **I**nterface.
- A **Web API** is an application programming interface for the Web.
- A **Browser API** can extend the functionality of a web browser.
- A **Server API** can extend the functionality of a web server.
- **Third party APIs** are not built into your browser. To use these APIs, you will have to download the code from the Web.

JavaScript Fetch API

HAZZA INSTITUTE OF TECHNOLOGY

The Fetch API interface allows web browser to make HTTP requests to web servers.

The basic syntax is:

```
1  let promise = fetch(url, [options])
```

- `url` – the URL to access.
- `options` – optional parameters: method, headers etc.

Without `options`, this is a simple GET request, downloading the contents of the `url`.

Example

JavaScript Fetch API (GET)

HAZZA INSTITUTE OF TECHNOLOGY

The Fetch API interface allows web browser to make HTTP requests to web servers.

A typical fetch request consists of two `await` calls:

```
let response = await fetch(url, options); // resolves with response headers
let result = await response.json(); // read body as json
```

Response properties:

- `response.status` – HTTP code of the response,
- `response.ok` – `true` is the status is 200-299.
- `response.headers` – Map-like object with HTTP headers.

```
let response = await fetch(url);

if (response.ok) { // if HTTP-status is 200-299
  // get the response body (the method explained below)
  let json = await response.json();
} else {
  alert("HTTP-Error: " + response.status);
}
```

Methods to get response body:

- `response.text()` – return the response as text,
- `response.json()` – parse the response as JSON object,
- `response.formData()` – return the response as `FormData` object (form/multipart encoding, see the next chapter),
- `response.blob()` – return the response as Blob (binary data with type),
- `response.arrayBuffer()` – return the response as ArrayBuffer (low-level binary data),

JavaScript Fetch API (POST)

HAZZA INSTITUTE OF TECHNOLOGY

The Fetch API interface allows web browser to make HTTP requests to web servers.

```
let user = {
  name: 'John',
  surname: 'Smith'
};

let response = await fetch('/article/fetch/post/user', {
  method: 'POST',
  headers: {
    'Content-Type': 'application/json;charset=utf-8'
  },
  body: JSON.stringify(user)
});

let result = await response.json();
alert(result.message);
```

Fetch (javascript.info)

Summary

Module 09

HAZZA INSTITUTE OF TECHNOLOGY

- DOM Model Understanding
- Understanding of methods document.getElementById()
- Understanding of methods document.getElementByClass Name()
- Understanding of methods document.getElementsByTag Name()
- Understanding of methods document.querySelector()
- Understanding of methods document.querySelectorAll()