Școala informală de IT

DOM Manipulation

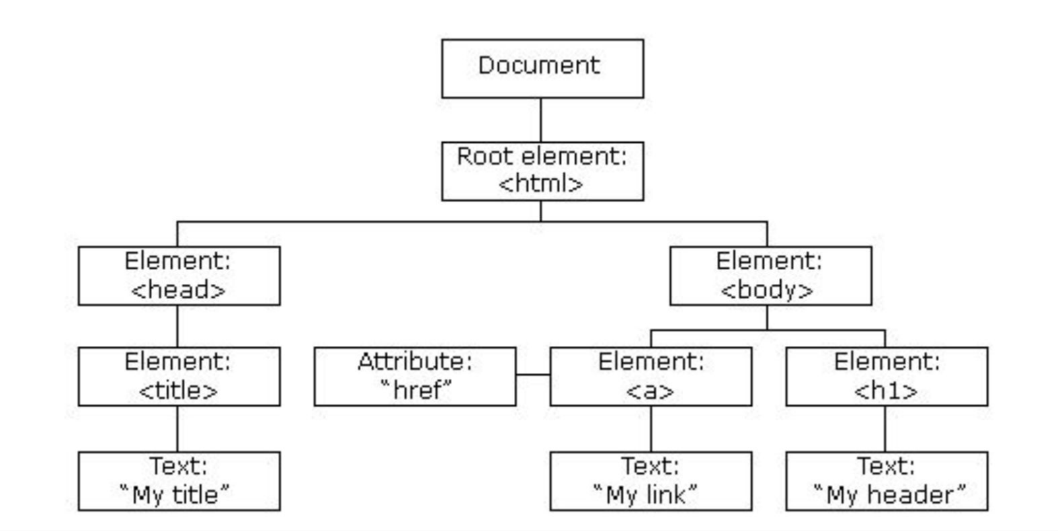
Agenda

- What the DOM is
- Safe DOM manipulation
- DOM manipulation
  - Finding elements, getting and setting content and attributes
  - CSS manipulation
  - Adding and Removing DOM elements
- Traversing the DOM
- Events

What the DOM is

The HTML DOM Tree of Objects

- **When a page is loaded, the browser creates a Document Object Model of the page.**
- **The HTML DOM model is constructed as a tree of objects:**



Document

Root element: <html>

Element: <head>

Element: <body>

Element: <title>

Attribute: "href"

Element: <a>

Element: <h1>

Text: "My title"

Text: "My link"

Text: "My header"

DOM

- **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

DOM (Summary)

- The Document Object Model (DOM) is a standard object model and programming interface (API) for HTML.
- It defines:
  - The HTML elements as objects
  - The properties of all HTML elements
    - = a value that you can get or set
  - The methods to access all HTML elements
    - = an action that you can do
  - The events for all HTML elements

The DOM Programming Interface

- The programming interface = the set of properties and methods.
- Document
  - If you want to access any element in an HTML page, always start with the document
- Window
  - The Browser Object Model (BOM) allows JavaScript to "talk to" the browser. All global JavaScript objects, functions, and variables automatically become members of the window object.
    - Global variables are properties of the window object.
    - Global functions are methods of the window object.
    - Even the document object (of the HTML DOM) is a property of the window object

Safe DOM Manipulation

Safe DOM Manipulation

- There are two main events that are important for DOM manipulation:
  - The DOM is ready and parsed. No images or external sources are loaded at this point
  - The DOM and the external sources are fully loaded

```
<script>
  document.addEventListener("DOMContentLoaded", function(event) {
    console.log("DOM fully loaded and parsed");
  });
</script>

<script>
  window.addEventListener("load", function(event) {
    console.log("All resources finished loading!");
 });
</script>
```

DOM Manipulation Methods

Finding elements in DOM

```
// by id
document.getElementById("myId");


// by class name
document.getElementsByClassName("myClass");


// by tag name
document.getElementsByClassName("myClass");


// by css selector; returns the first element that matches the selector
document.querySelector("#myid .myclass p");


// by css selector; returns a list of all elements that match the selector
document.querySelectorAll("#myid .myclass p");
```

Changing HTML elements (working with getters and setters)

```
// GET
// return the content for the selected element
document.getElementById("myId").innerHTML;


// return the value of the selected attribute for the selected element
document.getElementById("myId").getAttribute("title");


// SET
// updates the content for the selected element
document.getElementById("myId").innerHTML = "Some text";


// sets the value of the selected attribute for the selected element
document.getElementById("myId").setAttribute("title", "New title");
```

CSS Manipulation

```
// setting a css style
document.getElementById("my-button").style.background = "yellow";


// adding & removing css classes
var classes = document.getElementById("my-button").classList;
classes.add("new-class");
classes.remove("old-class");
```

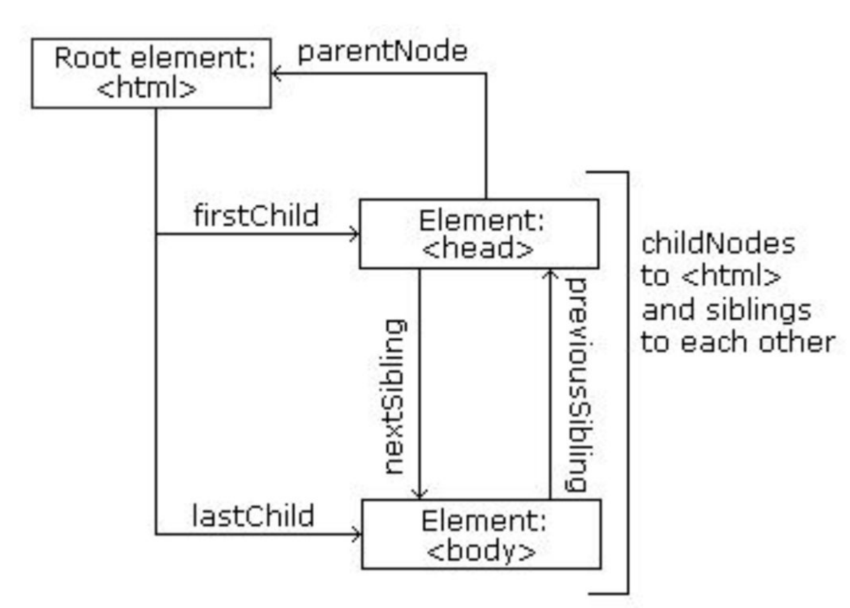
Adding / Removing DOM elements

```
// Remove DOM node
element.removeChild(child);


// Add DOM node

// 1. Create DOM element
var child = document.createElement("div");
// or text node
var textNode = document.createTextNode("New text node here");

// 2. Append the newly created element
element.appendChild(child);
// or text node
element.insertBefore(textNode, child);
// (both methods can be used on both types of nodes)


// Replace DOM node
element.replaceChild(newChild, oldChild);
```

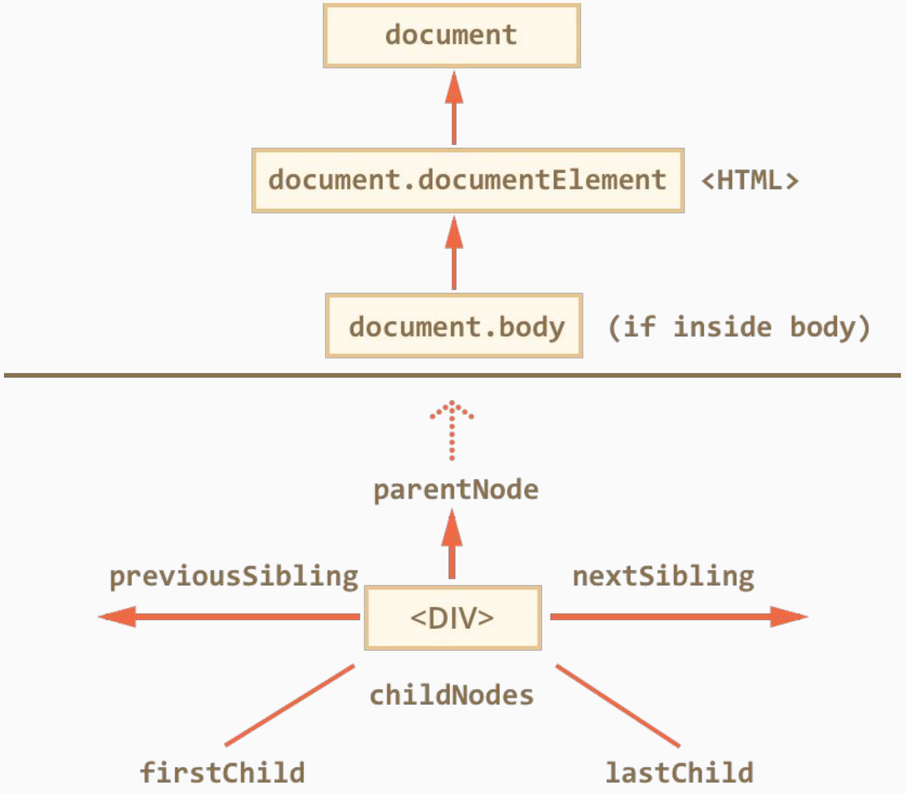
Traversing the DOM

Traversing the DOM

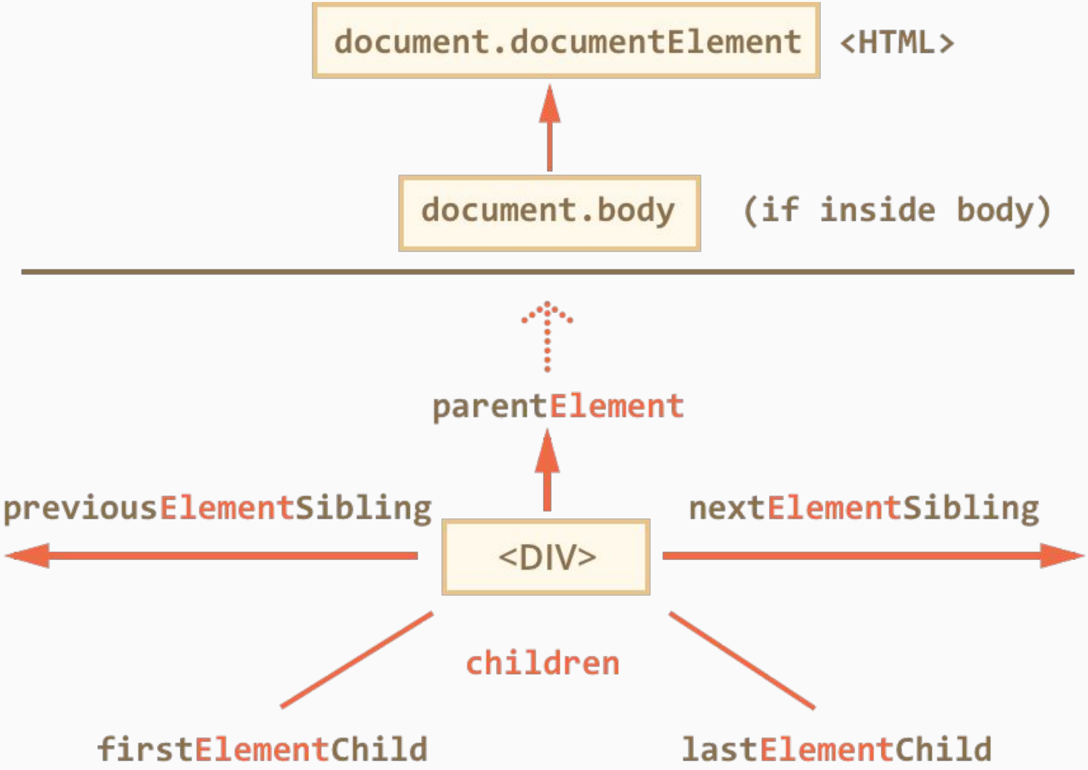
- **The nodes in the tree have a hierarchical relationship.**
- **The terms used to describe it are:**
    - **Parent**
    - **Child**
    - **Sibling**



Traversing the DOM (nodes)



Traversing the DOM (elements)



Events

Javascript Events - Examples

- When a user clicks the mouse
- When a web page has loaded
- When an image has been loaded
- When the mouse moves over an element
- When an input field is changed
- When an HTML form is submitted
- When a user strikes a key

Event Handling Methods

addEventListener

- attaches an event handler to the specified element, without overwriting existing event handlers
- can be used to add many event handlers to one element, even if they are of the same type (i.e. two "click" events)

```
document.getElementById("myId").addEventListener("click", function() {
  console.log("I was clicked");
});
```

removeEventListener

```
document.getElementById("myId").removeEventListener("click", function() {
  console.log("I was clicked");
});
```

Handling events

```
document.getElementById("myId").addEventListener("click", function() {
  console.log("I was clicked");
});

document.getElementById("myId").onclick = function() {
  console.log("I was clicked");
}

document.getElementById("myId").addEventListener("click", function(event) {
  console.log(event);
});

document.getElementById("myId").onclick = function(event) {
  console.log(event);
}
```

Resources

https://css-tricks.com/dom/

https://www.developeracademy.io/add-remove-css-classes-using-javascript/

http://javascript.info/dom-navigation

https://developer.mozilla.org/en-US/docs/Web/API/Node/removeChild

https://developer.mozilla.org/en-US/docs/Web/API/Node/appendChild

https://developer.mozilla.org/en-US/docs/Web/API/Document_Object_Model/Events