

JS & HTML

What is the DOM?

DOM (Document Object Model)

Programming interface that helps us to create, change or remove elements from a document. It allows to add events.

The DOM views an HTML document as a tree of nodes. A node represents an HTML element.



DOM tree structure

```
<!DOCTYPE html>
```

```
<html lang="en">
```

```
<head>
```

```
<meta charset="UTF-8">
```

```
<meta name="viewport" content="width=device-width, initial-scale=1.0">
```

```
<meta http-equiv="X-UA-Compatible" content="ie=edge">
```

```
<title>DOM tree structure</title>
```

```
</head>
```

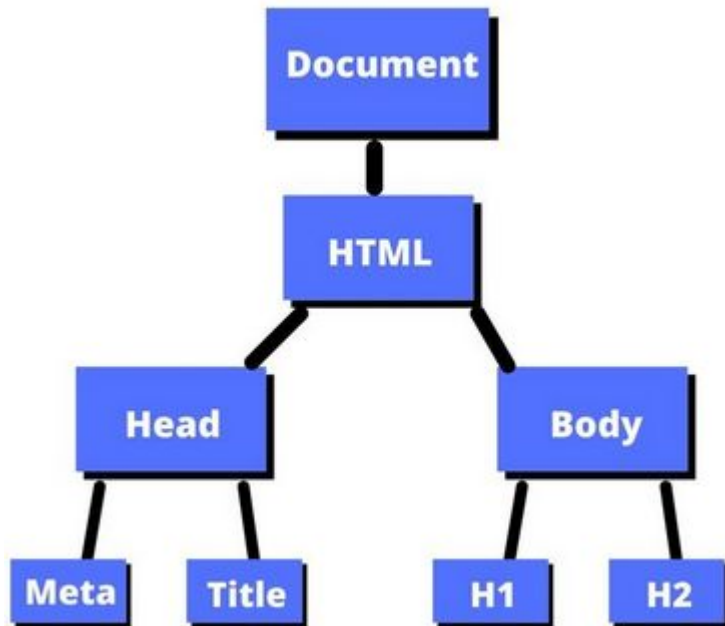
```
<body>
```

```
<h1>DOM tree structure</h1>
```

```
<h2>Learn about the DOM</h2>
```

```
</body>
```

```
</html>
```



Select Elements in the Document - getElementById()

getElementById(): returns an Element object representing the element whose id property matches the specified string.

```
<p id="p1">First paragraph.</p>
```

```
<p id="p2">Second paragraph.</p>
```

In JavaScript, we can grab an HTML tag by referencing the id name.

```
document.getElementById("id_here")
```

```
const paragraph1 = document.getElementById("p1");
```

```
console.log(paragraph1); => <p id="p1">First paragraph.</p>
```

Read the content of the paragraph:

```
const paragraph1 = document.getElementById("p1");
```

```
console.log(paragraph1.textContent); => "First paragraph."
```



Select Elements in the Document - `querySelector()`

`querySelector()`: returns the first Element within the document that matches the specified selector. If no matches are found, null is returned.

Find and print to the console the h1 element:

```
const h1Element = document.querySelector("h1");
```

With a class, use `.list` inside the `querySelector()` to target a `class="list"`:

```
const list = document.querySelector(".list");
```



Select Elements in the Document - `querySelectorAll()`

`querySelectorAll()`: Returns a `NodeList` representing a list of the document's elements that match the specified group of selectors.

To obtain a `NodeList` of all of the `<p>` elements in the document:

```
const matches = document.querySelectorAll("p");
```



DOM AND JS

The browser implements all the properties and methods that the DOM will need, and with your JS code you will be able to use them and modify the web page.




Let's add some JS code to our movies webpage

In your movies.html:

```
<body>
  <h1>Movie Theater</h1>
  <input type="text" class="seat_selector" maxlength="3" size="3" />
  ....
  .....
</body>
```

In the console

```
// Save seat input in a variable
seat_input = document.querySelector('.seat_selector')
// Apply this function
seat_input.getBoundingClientRect() => A DOMRect describes the size and position of a rectangle.
Seat_input.value => get the value of the element
seat_input.value = "345" => Modify the element
```



Events

With javascript we can create events. With an event, javascript knows it has to monitor an action, and when this action occurs, we can tell javascript to run a function.

Let's add a button in our html page.

```
<input
  type="text"
  class="seat_selector"
  maxlength="3"
  size="3"
/><br /><br />
<button type="button" class="button_check">Button</button>
```

In the JS file (movies.js):

```
"use strict"
document.querySelector(".button_check").addEventListener("click", function () {
  console.log("The button has been click");
});
```

If you click multiple times, the browser will display the message once, but there will be a number displayed on the right or left side (depending on the browser) of your console. The number tells how many consecutive times this message has been displayed.

Let's create another button.

```
<button type="button" class="button_check">Button</button><br /><br />  
<button type="button" class="button_css">My Second Button</button><br /><br />
```

In the javascript file, we are going to change the background color of the input element into red

```
document.querySelector(".button_css").addEventListener("click", function () {  
  document.querySelector(".seat_selector").style.backgroundColor = "red";  
});
```

So when we click on the button, the background color is going to be red.




Exercise

Let's do an exercise. Let's alternate the background color, one time red, one time clear.

Solution:

```
document.querySelector(".button_css").addEventListener("click", function () {  
  let input_element_style = document.querySelector(".seat_selector").style;  
  if (input_element_style.backgroundColor === "red") {  
    input_element_style.backgroundColor = "";  
  } else if (input_element_style.backgroundColor === "") {  
    input_element_style.backgroundColor = "red";  
  }  
});
```




We can also create a variable to store how many times we change the background.

```
<button type="button" class="button_css">  
  Change input background color</button><br /><br />  
<span class="bg_info">How many times have we changed the background? 0</span>
```



Then let's add some lines to the event:

```
let bg_change_times = 0; // Initialize a variable
document.querySelector(".button_css").addEventListener("click", function () {
  let input_element_style = document.querySelector(".seat_selector").style;
  if (input_element_style.backgroundColor === "red") {
    input_element_style.backgroundColor = "";
  } else if (input_element_style.backgroundColor === "") {
    input_element_style.backgroundColor = "red";
  }
  bg_change_times++;
  document.querySelector(
    ".bg_info"
  ).textContent = `We have change ${bg_change_times} times the input background color`;
});
```



Events using a class

Let's create a css

```
<link rel="stylesheet" href="main.css" />
```

And inside the css:

```
.hidden {  
  display: none;  
}
```

In the html:

```
<button type="button" class="button_class">Change element class</button><br /><br />  
<span class="hidden boo">I was hidden. Boooo!!!! Now you see me</span>
```

And in the javascript:

```
document.querySelector(".button_class").addEventListener("click", function () {  
  document.querySelector(".boo").classList.remove("hidden");  
});
```

We have removed the class "hidden" from the list of classes of the element.



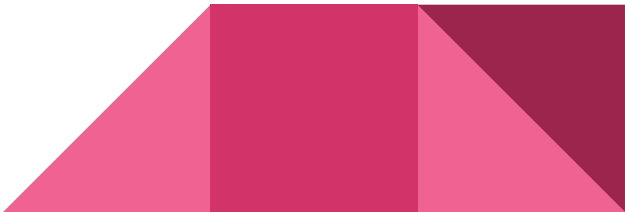
Events using a class - Exercise

Use the previous code and modify it to alternate show and hide.

Hint: Your solution will probably use a list and you will need to check if a certain element is in the list, use **`your_list.contains(item)`**

Solution:

```
document.querySelector(".button_class").addEventListener("click", function () {  
  let boo_class_list = document.querySelector(".boo").classList;  
  if (boo_class_list.contains("hidden")) {  
    boo_class_list.remove("hidden");  
  } else {  
    boo_class_list.add("hidden");  
  }  
});
```



Key Event

Keyboard event is a global event, because it cannot be assigned to an element. There is an event that listens to any keyboard key that has been pressed down.

It is:

```
document.addEventListener('keydown')
```

So let's try it

```
document.addEventListener('keydown', function () {  
    console.log('A key was pressed down')  
})
```

Even when you write in the input field, it is going to catch the event.



Key Event

Let's see if we can avoid logging into the console when the text input field is active.

In the javascript:

```
const seat_selector = document.querySelector(".seat_selector")
document.addEventListener('keydown', function () {
    if (document.activeElement !== seat_selector){
        console.log('A key was pressed down')
    }
})
```



Key Event

Let's filter which key can log into the console.

We can specify a variable to the function inside the event listener of 'keydown', which will match the event (https://developer.mozilla.org/en-US/docs/Web/API/Element/keydown_event)

```
document.addEventListener("keydown", function (event) {  
  if (document.activeElement != seat_selector) {  
    if (event.key == "r") {  
      console.log("A key was press", `${event.key}`);  
    }  
  }  
});
```

What is that code doing?

