

# מבוא ל JavaScript

קורס : צד לקוח

מרצה: יעל סלע זעירא

# היום בהרצאה

---

- 1. JS Intro רקע כללי
- 2. אירועים ופונקציות (מבוא)
- 3. Variables
- 4. Data Types
- 5. getElement
- 6. querySelector
- 7. DOM
- 8. parse

# Javascript Intro

---

javascript היא אחת משלושת שפות הליבה של האינטרנט HTML, CSS, JS  
מטרתה להפוך את דף ה HTML הסטטי לדף דינמי.

3 שכבות:

שכבת ה HTML

שכבת ה CSS

שכבת ה Javascript

דוגמא דף: JSIntro1.html

# Javascript Intro

---

- שפת תכנות עילית כמו C#/JAVA, אך מדובר בשפת script ולא בשפת קומפילציה.
- הקוד נקרא ומתחיל לרוץ מלמעלה למטה, לכן יש חשיבות לסדר, והתוצאה מייד מוצגת.
- הקוד רץ בצד הלקוח ב browser.

# Add Javascript code

---

```
<script>  
    // JavaScript goes here  
</script>
```

**OR**

```
<script src="script.js"></script>
```

# < script >

---

תג script יכול להופיע בתג head ו\או בתג body.

```
<html>
<head>
  <title></title>
  <meta charset="utf-8" />
  <script>
    //code block 1
  </script>
</head>
<body>

  <script>
    //code block 2
  </script>

</body>
</html>
```

# alert

---

לשפה יש פונקציות מובנות, הראשונה שנכיר זו פונקציית alert שמקפיצה חלון התראה במסך.

```
<head>  
  <script>  
    alert("Hello from javascript")  
  </script>  
</head>
```

# Avoiding alert

---

1. Alert עוצר את הרצת ה javascript עד אשר המשתמש משחרר את ה dialog
2. לא ידידותי למשתמש
3. חלון מיושן והעיצוב עברו דל

**לצורכי debug עדיף להשתמש ב `console.log`**



# Avoiding alert

---

כאשר בכל זאת רוצים alert כדאי להשתמש בכל modal אחר למשל `<dialog>`

```
<dialog id="myDialog">This is a modern alert!</dialog>
```

```
<script>
```

```
  const dialog = document.getElementById('myDialog');
```

```
  dialog.showModal();
```

```
  setTimeout(() => dialog.close(), 2000); // Auto-close after 2 seconds
```

```
</script>
```

# console.log

---

```
<head>  
  <script>  
    console.log("Hello from javascript")  
  </script>  
</head>
```

# function

---

לפונקציות ב-javascript אין טיפוס החזרה. הגדרת פונקציה נעשית באמצעות המילה

השמורה `function`

```
//func is the name of the function
function func()
{
    //code goes here
}
```

```
func(); //use () on the function name to activate the function
```

# function

בתוך תג script ניתן לשים קטעי קוד רבים, בד"כ בתג יהיו הרבה פונקציות שיחווטו לפעולות שיבצע המשתמש.

קטע הקוד לא יציג את ה alert

```
<script>

  function userClickAlert()
  {
    alert("user click!");
  }

</script>
```

קטע הקוד יציג את ה alert

```
<script>

  function userClickAlert()
  {
    alert("user click!");
  }
  userClickAlert();
</script>
```

# event

הדרך הפשוטה ביותר לחוות אלמנט לפונקציה, היא דרך הוספת **תכונת אירוע לאלמנט** עצמו:

```
<html>
<head>
  <title></title>
  <meta charset="utf-8" />
  <script>
    function userClickAlert()
    {
      alert("user click!");
    }
  </script>
</head>
<body>
  <input type="button" value="click me" onclick="userClickAlert()"/>
</body>
</html>
```

יש עוד רשימת אירועים גדולה

שנכיר בהמשך

ודרכים נוספות לחוות אירועים.

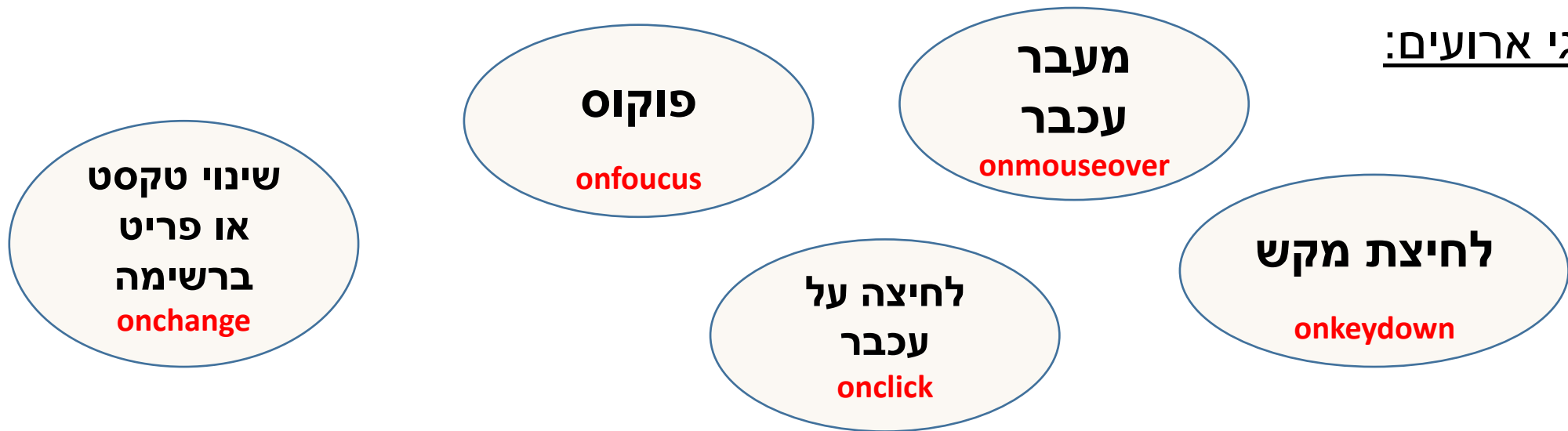
# תכנות מונחה אירועים

---

אתרי האינטרנט בנויים על אירועים שמייצר המשתמש.

מרבית דפי האינטרנט שתגיעו אליהם, לא יעשו כלום כל עוד המשתמש לא ביצע דבר.

סוגי אירועים:



# חיבור אירוע לאלמנט

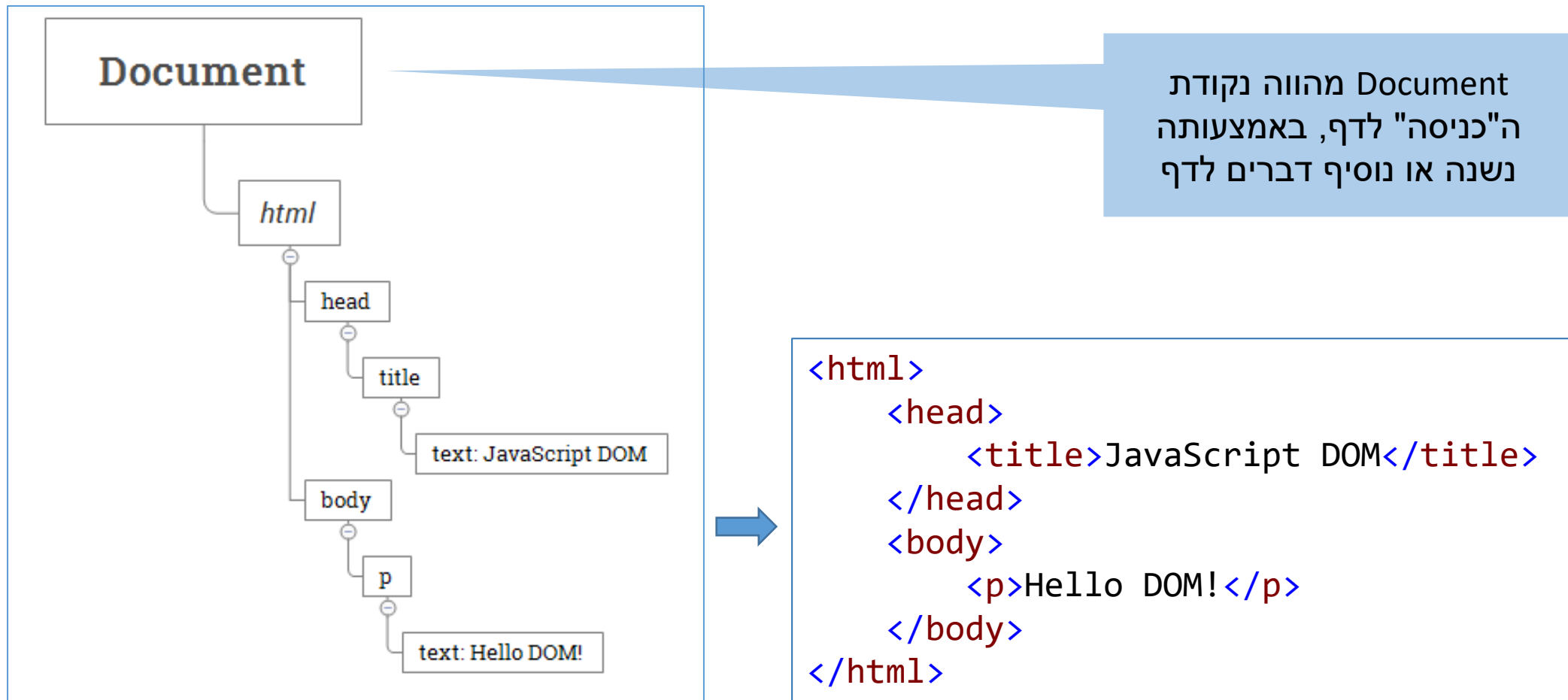
- חיבור אירוע לאלמנט יכול להתבצע בצורה סטטית או דינאמית (היום אנו רואים רק את הצורה הסטטית).
- החיבור של לחיצת כפתור שביצענו עד כה הוא דוגמא לחיבור סטטי.
- יש אלמנטים שלא תומכים באירועים מסוימים, ובמצב כזה האירוע פשוט לא יופעל.

```
<input id="btn" type="button" value="change to red color" onclick="changeColor()"/>
```

חיבור סטטי

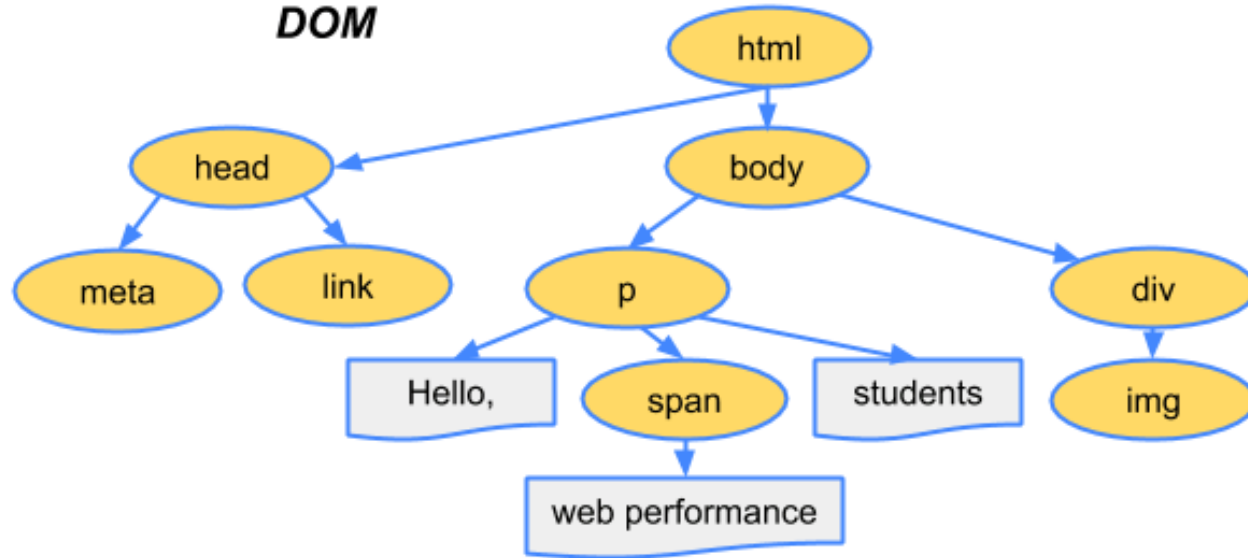
# DOM – Document object model

מודל אותו הבראוזר מייצר טרם עליית הדף. המודל מייצג את כל האלמנטים שנמצאים בדף ה html

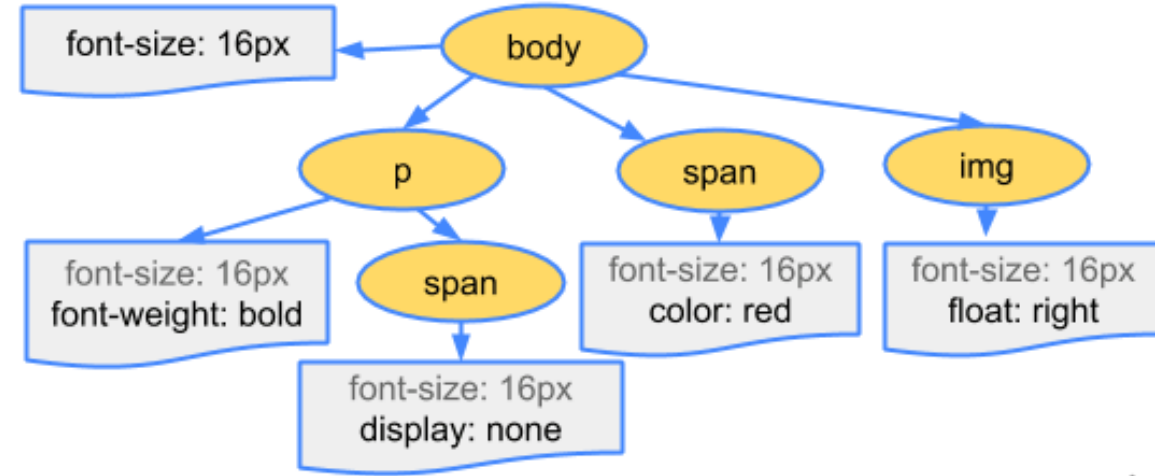




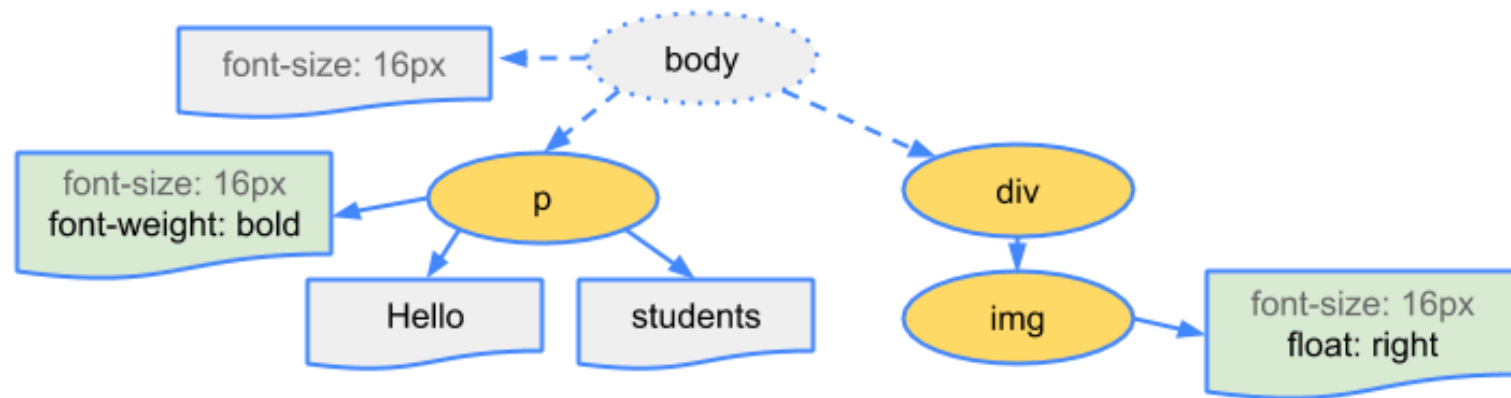
## DOM



## CSSOM



## Render Tree



# גישה לאלמנט

---

- אחת הפונקציות בגישה לאלמנטים היא `getElementById`

```
document.getElementById("someId")
```

- מביאה אלמנט html לפי ה-id שלו, כך ניתן לגשת לתכונות האלמנט ולשנות את ערכם.
- id של אלמנט הוא `unique`.
- אם ה-id לא קיים מוחזר `null`.

# שינוי אלמנט

---

נניח ובלחיצה על כפתור נרצה לשנות את טקסט הכפתור – הכניסו לפונקציה:

```
<html>
<head>
  <title></title>
  <meta charset="utf-8" />
  <script>
    function userClick()
    {
      document.getElementById("btn").value = "user click me!";
    }
  </script>
</head>
<body>
  <input type="button" value="click me" onclick="userClick()" id="btn"/>
</body>
</html>
```

# שינוי אלמנט

---

נניח ובלחיצה על כפתור נרצה לשנות את צבע הטקסט של הכפתור

style זו תכונה מורכבת (אובייקט) שיש לה תכונות נוספות.

```
<html>
<head>
  <title></title>
  <meta charset="utf-8" />
  <script>
    function changeTextColor()
    {
      document.getElementById("btn").style.color = "red";
    }
  </script>
</head>
<body>
  <input type="button" value="change my text color" onclick="changeTextColor()" id="btn"/>
</body>
</html>
```

# onmouseover, onmouseout

---

שינוי צבע כפתור לפי מעבר עם עכבר ויציאה עם עכבר (onmouseover, onmouseout):

```
<input id="btn" type="button" value="hover me" onmouseover="setColor()" onmouseout="resetColor()"/>

<script>
  function setColor()
  {
    document.getElementById("btn").style.color = "red";
  }

  function resetColor()
  {
    document.getElementById("btn").style.color = "";
  }
</script>
```

# onfocus, onblur

---

שינוי צבע תיבת טקסט כאשר בפוקוס או ביציאה מהתיבה (העכבר בקליק על תיבת הטקסט - onfocus, onblur):

```
<input id="btn" type="text" onfocus="setBackColor()" onblur="resetBackColor()"/>

<script>
  function setBackColor()
  {
    document.getElementById("btn").style.backgroundColor = "red";
  }

  function resetBackColor()
  {
    document.getElementById("btn").style.backgroundColor = "";
  }
</script>
```

# onchange

---

שינוי פריט ברשימה נגללת (select....option) - עשו alert עם שינוי (onchange):

```
<select onchange="doSomething()">
  <option>1</option>
  <option>2</option>
  <option>3</option>
</select>
```

# onkeyup

ברגע עזיבה של מקש מתיבת טקסט שנו את הצבע של הכפתור (onkeyup):

```
<input id="btn" type="text" onkeyup="setBackColor()"/>

<script>
  function setBackColor()
  {
    var x = document.getElementById("btn");
    x.style.backgroundColor = x.value;
  }
</script>
```

//add css class

```
function myFunction() {
  var element = document.getElementById("btn");
  element.classList.add("mystyle");
}
```

אירועים נוספים בסגנון:

onkeydown – ברגע שכפתור נלחץ (כל כפתור ירוץ במעגלים עד עזיבת המקש)

onkeypress – ברגע שכפתור נלחץ שמסמל תו (ירוץ במעגלים עד עזיבת המקש)

לרשימת אירועים נוספים



# this

---

לפעמים נרצה להעביר לפונקציות את כל האובייקט שלחצו עליו:

```
<input type="text" id="txt" />
<input type="button" value="btn" id="happy" onclick="f1(this)" />

<script>
  function f1(cntrl) {
    document.getElementById("txt").value = cntrl.id;
  }
</script>
```

בשביל לדעת תכונות מסויימת שיש לאותו האובייקט שנלחץ.

או בכללי לדעת על מי לחצנו.

# this

---

```
<html>
```

```
<script>
```

```
function colorChanger(e1) {  
    e1.style.backgroundColor = '#007d00';  
}
```

```
</script>
```

```
<body>
```

```
<h2>The JavaScript <i>this</i> Keyword</h2>
```

```
<button onclick="colorChanger(this)">Click to change Me!</button>
```

```
</body>
```

```
</html>
```

# תרגיל כיתה 1

(1) צרו 2 כפתורי רדיו left ו right, כל לחיצה תזיז את כותרת הדף בהתאם (רק אחד יכול להיות מסומן).

(2) צרו תיבת טקסט וכפתור, כאשר בלחיצה על הכפתור **יעבור**

הטקסט שכתוב בתיבה לטקסט הכתוב על הכפתור (יעבור=יימחק מהתיבה).



The screenshot shows a web browser window with a form titled "header". Inside the form, there are two radio buttons: "left" (which is selected) and "right". Below the radio buttons is a text input field. At the bottom of the form is a button labeled "button".

jsIntroEx1.html

# variables

```
<head>
  <title></title>
  <meta charset="utf-8" />
  <script>

    function userClick()
    {
      document.getElementById("btn").value = "user click me!";
      document.getElementById("btn").style.color = "red";
    }

  </script>
</head>
<body>

  <input type="button" id="btn" value="click me" onclick="userClick()" />

</body>
```

- ביצועים – זמני ריצה ארוכים יותר
- קוד ארוך ולא נקי

# variables

---

- המילה השמורה **let** מגדירה משתנה

- ב-javascript **אין הגדרה לטיפוס** - משתנה הוא מיכל לשמירת ערכים

- Javascript היא case sensitive : $x \neq X$

\*ניתן להגדיר משתנה גם עם **var** ו-**const** נראה בהמשך

# let

---

הגדרת משתנה מתבצעת על ידי המילה **let**

```
let message;
```

לדוגמא – הצהרה על המשתנה message :

```
let message;  
message = 'Hello!';  
alert(message); // shows the variable content
```

כעת נשים לתוכו ערך :

```
let message = 'Hello!'; // define the variable and assign the value  
alert(message); // Hello!
```

# let

```
let user = 'John', age = 25, message = 'Hello';
```

עדיף בצורה הבאה: ארוך אבל יותר קריא



```
let user = 'John';  
let age = 25;  
let message = 'Hello';
```



```
let message = "This";  
// repeated 'let' leads to an error  
let message = "That"; // SyntaxError: 'message' has already been declared
```



לא ניתן להצהיר על אותו משתנה פעמיים

# שמות משתנים

שתי מגבלות:

1. השם חייב להכיל אותיות או/ו מספרים או/ו את הסימנים \$, \_

2. האות הראשונה חייבת להיות **לא** ספרה

דוגמאות:

```
let userName;  
let test123;  
let $ = 1; // declared a variable with the name "$"  
let _ = 2; // and now a variable with the name "_"
```


```
alert($); // 1  
alert(_); // 2
```

\*הגדרת משתנה תעבוד גם ללא המילה *let* – אך זה תרגיל גרוע



---

## הגדרות שגויות:

 `let 1a; // cannot start with a digit`  
`let my-name; // hyphens '-' aren't allowed in the name`

## סוג משתנים

```
let a = 1;           //number
let b = 1.4;         //number
let c = "ff";        //string
let d = 'gg';        //string
let e = [];          //array (the type is object)
let f = {};          //object
let g = true;        //boolean
let h = null;         //null (the type is object)
let i;               //undefined
let l = alert;       //function
```

```
let a = 1;
alert(typeof a); // number
```

# undefined and null

---

משתנים שהוכרזו אך לא אותחלו או שאותחלו ב undefined  
יתנהגו בצורה הבאה:

- 1) **typeof** returns string 'undefined'
- 2) **==** check with null returns true
- 3) **==** check with undefined returns true
- 4) **===** check with null returns false
- 5) **===** check with undefined returns true

# undefined and null

---

משתנים שאותחלו בערך null

- 1) **typeof** returns string **'object'**
- 2) **==** check with null returns true
- 3) **==** check with undefined returns true
- 4) **===** check with null returns true
- 5) **===** check with undefined returns false

# constatnts

---

הגדרת קבוע על ידי המילה `const`

המשתנה – לא ניתן לשינוי

```
const myBirthday = '18.04.1982';
```

```
myBirthday = '01.01.2001'; // error, can't reassign the constant!
```

# uppercase constants

נהוג לתת לערכים שקשה לזכור והם קבועים וידועים מראש, שמות ב upper case.

כלומר ערכי hard-coded שידועים ניתן להם שמות uppercase

```
const COLOR_RED = "#F00";  
const COLOR_GREEN = "#0F0";  
const COLOR_BLUE = "#00F";  
const COLOR_ORANGE = "#FF7F00";  
  
// ...when we need to pick a color  
let color = COLOR_ORANGE;  
alert(color); // #FF7F00
```

יתרונות:

1. קל לזכור ולהשתמש שוב
2. פחות טעויות
3. שם משמעותי ומובן

# כללי אצבע לשמות משתנים

---

1. שמות מובנים וקריאים למשל : `userName`, `shoppingCart`
2. להימנע ממשתנים שקוראים להם : `a`, `b`, `c....`
3. שמות בעלי משמעות למשל השמות : `data`, `value` לא באמת אומרים לנו מהו המשתנה
4. שיחזור משתנים, אין צורך, הגדירו משתנה חדש הבראזרים היום מהירים.

## תרגיל כיתה 2

---

### כתוב פונקציה בשם `swap`

- הפונקציה מגדירה שני משתנים  $x, y$  עם הערכים 111 ו 333
- מבצעת להם `swap`
- מדפיסה לקונסול את  $x$  ו  $y$
- וודא שהערכים התחלפו



# שאלה

---

איזה מהשמות של הקבועים הבאים היית משנה אם בכלל?

```
const BIRTHDAY = '18.04.1982'; // uppercase OK?
```

```
const AGE = someCode(BIRTHDAY); // uppercase OK?
```

פתרון:

כיוון ש age מחושב ב runtime עדיף יהיה להגדיר אותו באותיות קטנות

# פעולות מתמטיות על משתנים

---

פעולות מתמטיות על טיפוסים מסוג number מתנהגות כרגיל.

פעולות מתמטיות על טיפוסים מסוג string - כאשר מדובר במספר - מתנהגות כרגיל, להוציא פעולות חיבור.

```
"5" - "5"; //equal 0  
"5" / "5"; //equal 1  
"5" * "5"; //equal 25  
"5" + "5"; //equal 55
```

```
"5" - 5; //equal 0  
"5" / 5; //equal 1  
"5" * 5; //equal 25  
"5" + 5; //equal 55
```

חיבור מחרוזות בין עצמן או עם מספר – משרשר את הערכים למחרוזת אחת:

\*אם נבצע פעולות מתמטיות בין שני משתנים, תרגול טוב יהיה להפוך מחרוזת למספר לפני ביצוע הפעולה

(<https://dev.to/sanchithasr/7-ways-to-convert-a-string-to-number-in-javascript-4l>)

```
"hello " + 10;    //??
```

hello 10

---

```
"hello " + 10 + 5; //??
```

hello 105

---

```
10 + 5 + " hello"; //??
```

15 hello

---

```
<script>
// no error
let message = "hello";
message = 123456;
//number
let n = 123;
n = 12.345;
//special numeric values- Infinity, -
Infinity and NaN.
alert(1 / 0); // Infinity
alert(Infinity); // Infinity
alert("not a number" / 2); // NaN, such
division is erroneous

alert(NaN + 1); // NaN
alert(3 * NaN); // NaN
alert("not a number" / 2 - 1); // NaN
let str = "Hello";
let str2 = 'Single quotes are ok too';
let phrase = `can embed another ${str}`;
```

```
let name = "John";
// embed a variable
alert(`Hello, ${name}!`); // Hello, John!
// embed an expression
alert(`the result is ${1 + 2}`); // the result is
3
let nameFieldChecked = true; // yes, name field is
checked
let ageFieldChecked = false; // no, age field is
not checked
let isGreater = 4 > 1;
alert(isGreater); // true (the comparison result
is "yes")
let age = 100;
// change the value to undefined
age = undefined;
alert(age); // "undefined"
</script>
```

# תוכן אלמנט

```
<input />  
<textarea></textarea>    document.getElementById("inputId").value;
```

```
<div></div>  
<h1></h1>  
<span></span>  
<p></p>  
<a></a>  
<ul>  
    <li></li>  
</ul>    document.getElementById("divId").innerHTML;
```

# תרגיל כיתה 3

---

**צרו מונה שסופר קליקים באמצעות כפתור**

לחיצה על הכפתור:

- מעלה את המונה ב 1

Plus One

0

Minus One

**צרו כפתור נוסף שמוריד קליקים מהמונה**

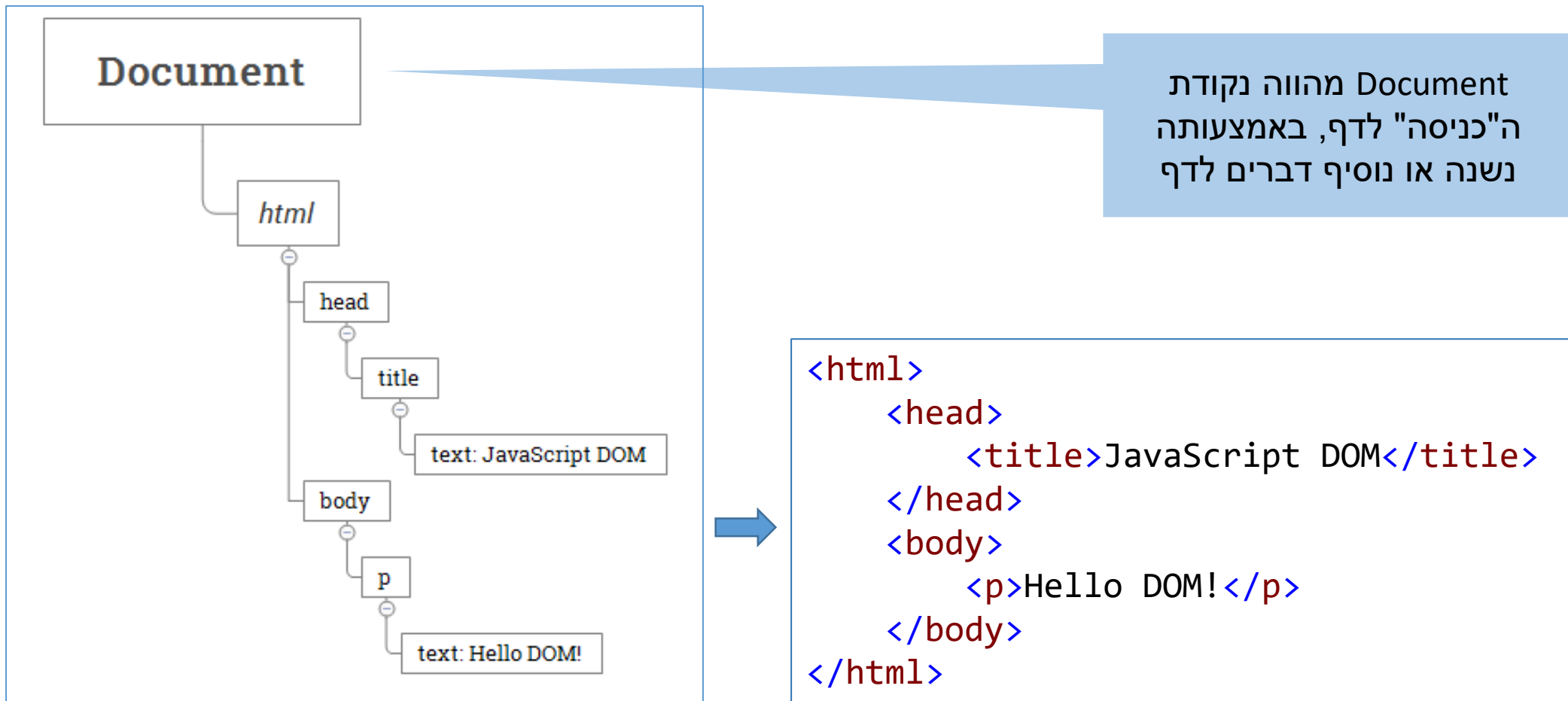
לחיצה על הכפתור:

- מוריד את המונה ב 1

jsBasicsClickCounter.html

# DOM – Document object model

מודל אותו הבראוזר מייצר טרם עליית הדף. המודל מייצג את כל האלמנטים שנמצאים בדף ה html



# DOM, CSSOM, Script

---

מנוע הקריאה של מסמך HTML ועיבודו לאובייקטים DOM ו-CSSOM מתחיל מהשורה הראשונה בדף ועד השורה האחרונה.

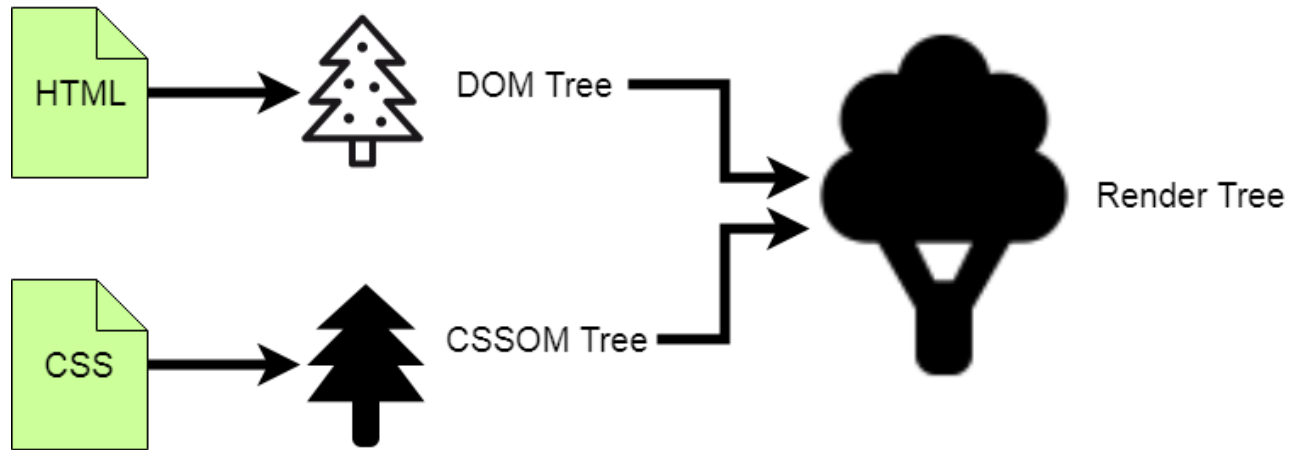
כאשר ייתקל בקובץ js או תג script – ייעצר כל תהליך הקריאה של המסמך עד שיושלם

הקוד



# שילבי טעינת דף HTML

## הדפדפן יוצר:

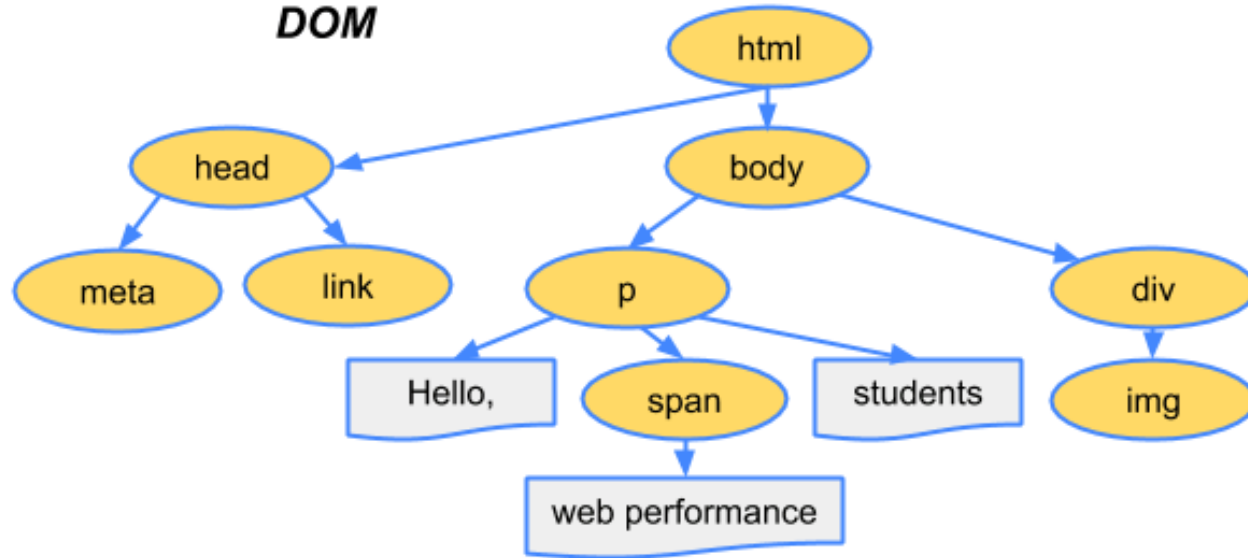


DOM עבור האלמנטים לתצוגה

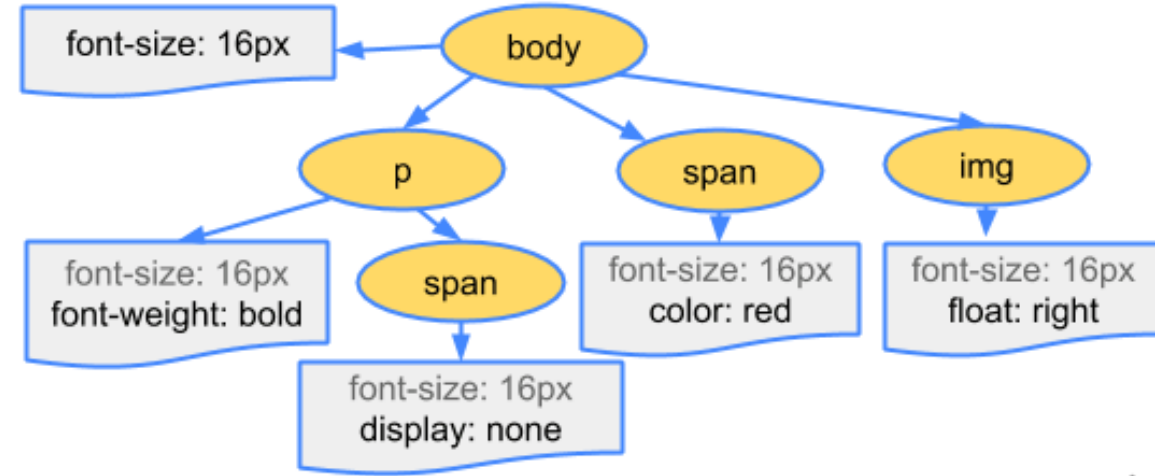
CSSOM עבור קבצי ה CSS

כאשר שני האובייקטים הללו יהיו מוכנים – יחל שלב עיבוד תצוגת המידע למשתמש.

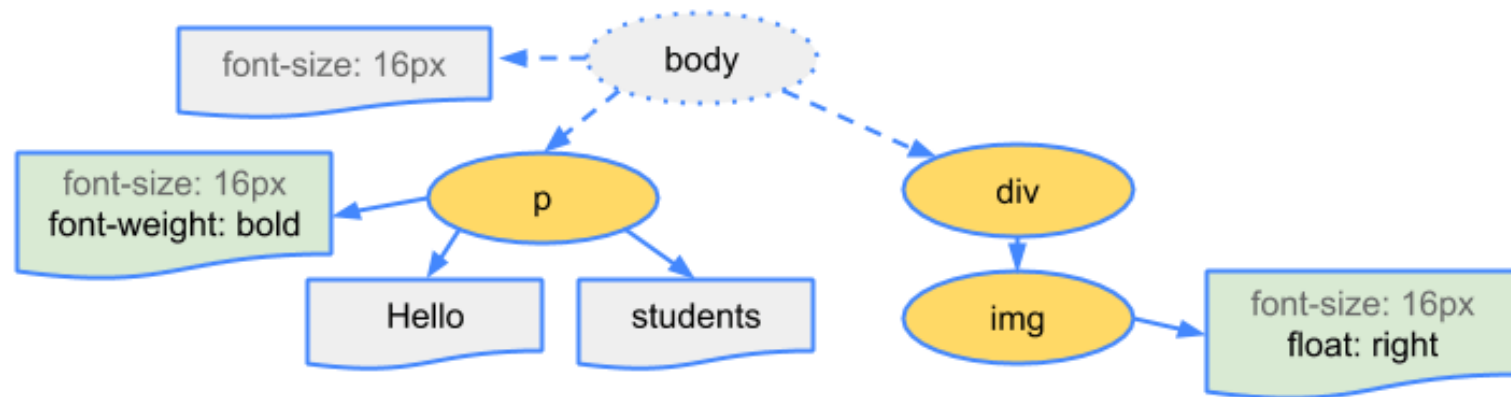
## DOM



## CSSOM



## Render Tree



# היכן נשים <script>?

---

1. בתוך ה <head>

2. לפני הסוגר של ה </body>

```
<head>
  <script>
    document.getElementById("num1").value;
  </script>
</head>
<body>
  <input type="text" id="num1"/>
</body>
```

✖ Uncaught TypeError: Cannot read properties of null (reading 'value')  
at classEx1.html:9

# שלבי טעינת דף HTML

---

לדף HTML יש 3 מצבים בהם הוא נמצא:

1. Loading – המסמך בשלב טעינה וקריאת מידע (עיבוד)
  2. Interactive – המסמך סיים להיטען מבחינת אובייקטים DOM ו-CSSOM, בשלב זה למשל ייתכן וקראנו תג `img`, אך לא בהכרח הוצגה התמונה
  3. Complete – המסמך מוכן לחלוטין כולל משאבים חיצוניים כמו תמונה, סרט וכו'..
- באמצעות התכונה **defer** ניתן להעביר script חיצוני שירוך בסיום שלב interactive (עובד רק על script חיצוני)

# getElementById

---

כפי שראינו אם יש לאלמנט id ניתן לגשת אליו על ידי getElementById או על ידי ה id

ה id חייב להיות unique אחרת יוחזר ערך רנדומלי

```
<div id="elem">
<div id="elem-content">Element</div>
</div>

<script>
// get the element
let elem = document.getElementById('elem');
// make its background red
elem.style.background = 'red';
</script>
```

Javascript מיצר משתנים גלובלים לאלמנטים עם id ולכן ניתן גם לגשת ישירות

```
<head>
</head>
<body>
  <div id="someElement">getElementById works</div>
  <div id="anotherElement">This also works?!</div>

  <script>
    var someElement = document.getElementById("someElement");
    console.log(someElement.innerHTML); // getElementById works
    console.log(anotherElement.innerHTML); // This also works?!
  </script>
</body>
```

לא משתמשים ישירות ב id בעיקר על מנע להימנע מטעויות – לעוד מידע מוזמנים לקרוא בלינק :  
<https://www.sitepoint.com/community/t/should-we-use-getelementbyid/279268>



`id="elem-content"    window['elem-content']`

---

```
<div id="elem">
<div id="elem-content">Element</div>
</div>

<script>
// elem is a reference to DOM-element with id="elem"
elem.style.background = 'red';

// id="elem-content" has a hyphen inside, so it can't be a variable name
// ...but we can access it using square brackets: window['elem-content']
</script>
```



# querySelector

# querySelectorAll

elem.querySelectorAll(css selector) מחזירה את כל האלמנטים בתוך elem שתואמים ל css סלקטור.

```
<body>
  <p>1</p>
  <p>2</p>
  <p>3</p>
  <p>4</p>
  <p>5</p>
  <div class="note">first div</div>
  <div class="note">second div</div>
  <div class="alert">alert div</div>
  <span id="#test">test span</span>
  <div class="highlighted"><p>highlighted p</p></div>
  <script>
    //all of the <p> elements in the document:
    let x1 = document.querySelectorAll("p");
    //class of either note or alert
    let x2 = document.querySelectorAll("div.note, div.alert");
    //inside a container whose ID is test - list of <p> elements whose immediate parent element is a <div>
    //with the class highlighted.
    let x3 = document.querySelector("#test");
    let x4 = x3.querySelectorAll("div.highlighted > p");
  </script>
</body>
```

# querySelector

---

יחזיר את האלמנט הראשון ב css לפי הסלקטור, במילים אחרות יוחזר:

```
elem.querySelectorAll(nameOfCssSeector)[0]
```

לכן, אם אנו רוצים את הראשון ורוצים אחד בלבד נשתמש ב querySelector שהוא יותר מהיר.

```
<html>
```

```
<body>
```

```
  <h1>The Document Object</h1>
```

```
  <h2>The querySelector() Method</h2>
```

```
  <p>Add a background color to the first element with class="example":</p>
```

```
  <p class="example">I am a paragraph.</p>
```

```
  <p class="example">I am a paragraph.</p>
```

```
  <script>
```

```
    document.querySelector(".example").style.backgroundColor = "red";
```

```
  </script>
```

```
</body>
```

```
</html>
```

# getElementsByTagName\*

---

getElementsByTagName – חיפוש לפי תג

getElementsByClassName – חיפוש לפי class

שתי הפונקציות מחזירות  
**אוסף**  
אפילו אם מדובר באלמנט אחד

```
<table id="table">
<tr>
  <td>Your age:</td>
<td>
<label>
<input type="radio" name="age" value="young" checked> less than 18
</label>
<label>
<input type="radio" name="age" value="mature"> from 18 to 50
</label>
<label>
<input type="radio" name="age" value="senior"> more than 60
</label>
</td>
</tr>
</table>
```

Your age: ☒ less than 18 ☐ from 18 to 50 ☐ more than 60

```
<script>
let inputs = table.getElementsByTagName('input');
for (let input of inputs) {
  console.log(input.value + ': ' + input.checked);
}
</script>
```

Young: true, mature:false, senior:false

# סיכום

Method	Searches by...	Can call on an element?
<code>querySelector</code>	CSS-selector	✓
<code>querySelectorAll</code>	CSS-selector	✓
<code>getElementById</code>	<code>id</code>	-
<code>getElementsByName</code>	name	-
<code>getElementsByTagName</code>	tag or <code>'*'</code>	✓
<code>getElementsByClassName</code>	class	✓

```

<div>
  Search the site:
  <input type="text">
  <input type="button" value="Search!">
</div>
<div id="search-person">
  Search the visitors:
  <table id="age-table">
    <tr>
      <td>Age:</td>
      <td id="age-list">
        <input type="radio" name="age" value="young">less than 18
        <input type="radio" name="age" value="mature">18-50
        <input type="radio" name="age" value="senior">more than 50
      </td>
    </tr>
    <tr>
      <td>Additionally:</td>
      <td>
        <input type="text">
        <input type="text">
        <input type="text">
      </td>
    </tr>
  </table>
  <input type="button" value="Search!">
</div>

```

## תרגיל כיתה 4

העתיקו את הקוד הבא אליכם - הדפיסו ל log את האלמנטים

1. מצאו את ה table בעל id="age-table"

2. כל האלמנטים מסוג input בתוך ה table הזה

3. ה td הראשון בתוך ה table הזה

4. כל האלמנטים מסוג input שבתוך הdiv עם id : search-person



# String conversion

---

```
let value = true;  
alert(typeof value); // boolean
```

```
value = String(value); // now value is a string "true"  
alert(typeof value); // string
```

# Numeric conversion

---

```
alert("6" / "2"); // 3, strings are converted to numbers
```

```
let str = "123";  
alert(typeof str); // string
```

```
let num = Number(str); // becomes a number 123
```

```
alert(typeof num); // number
```

---

```
let age = Number("an arbitrary string instead of a number");  
  
console.log(age); // NaN, conversion failed  
  
console.log(Number("  123  ")); // 123  
  
console.log(Number("123z")); // NaN (error reading a number at "z")  
  
console.log(Number(true)); // 1  
  
console.log(Number(false)); // 0
```

# parseInt()

---

- |                           |        |
|---------------------------|--------|
| a) parseInt("10");        | a) 10  |
| b) parseInt("10.00");     | b) 10  |
| c) parseInt("10.33");     | c) 10  |
| d) parseInt("34 45 66");  | d) 34  |
| e) parseInt(" 60 ");      | e) 60  |
| f) parseInt("40 years");  | f) 40  |
| g) parseInt("He was 40"); | g) Nan |
| h) parseInt("010") ;      | h) 10  |

# המרות למספרים - שימוש בקוד

```
<html>
<head>
  <title></title>
  <meta charset="utf-8" />

</head>
<body>
  <input type="button" value="conuter" onclick="counterAdd()" />
  <span id="spn">0</span>

  <script>
    var x = document.getElementById("spn");

    function counterAdd()
    {
      x.innerHTML = parseInt(x.innerHTML) + 1;
    }
  </script>
</body>
</html>
```

נביט בתרגיל הכיתה של ה counter:

# Boolean conversion

---

```
alert(Boolean(1)); // true
```

```
alert(Boolean(0)); // false
```

```
alert(Boolean("hello")); // true
```

```
alert(Boolean("")); // false
```

# ספריית Math

---

ספרייה המשמשת ליצירת מספרים אקראיים, עיגול מספר דצימלי, ערך מוחלט ועוד.

`<script>`

```
Math.abs(-3);    //return 3
Math.ceil(3.3);  //return 4 - rounded up
Math.floor(3.7); //return 3 - rounded down
Math.round(3.3); //return 3 - rounded to nearest
Math.random();   //return random number between 0 to 1
Math.min(1, -1, 3); //return -1 the minimum number
Math.max(1, -1, 3); //return 3 the maximum number
```

`</script>`

# הצגת האינדקס והערך של ה current selected value

---

```
var x = document.getElementById("mySelect");
```

```
document.getElementById("mySelect").selectedIndex = "2";
```

```
Console.log("Index: " + x[x.selectedIndex].index + " is " + x[x.selectedIndex].text);
```

האינדקס בבסיס 0

optionExample.html



## תרגיל כיתה 5

---

צרו מחשבון ע"י 2 תיבות טקסט בהן יזין המשתמש ערכים (הניחו שמדובר במספרים).

בין תיבות הטקסט תהיה רשימה נגללת שמכילה 2 פעולות חשבון - מינוס ופלוס (השתמשו ב `if ==`).

לאחר התיבות יהיה כפתור "=" ו-`span` שמציג את התוצאה.

<input type="text" value="5"/>	<input type="text" value="+ ▼"/>	<input type="text" value="7"/>	<input type="button" value="="/> 12
--------------------------------	----------------------------------	--------------------------------	-------------------------------------

ברגע לחיצה על הכפתור, על סמך הפעולה ברשימה יתבצע החישוב.