

# JS2

# היום בהרצאה

---

1. Conditions – if .... else

2. אופרטורים לוגיים

3. לולאות for, while

4. Switch

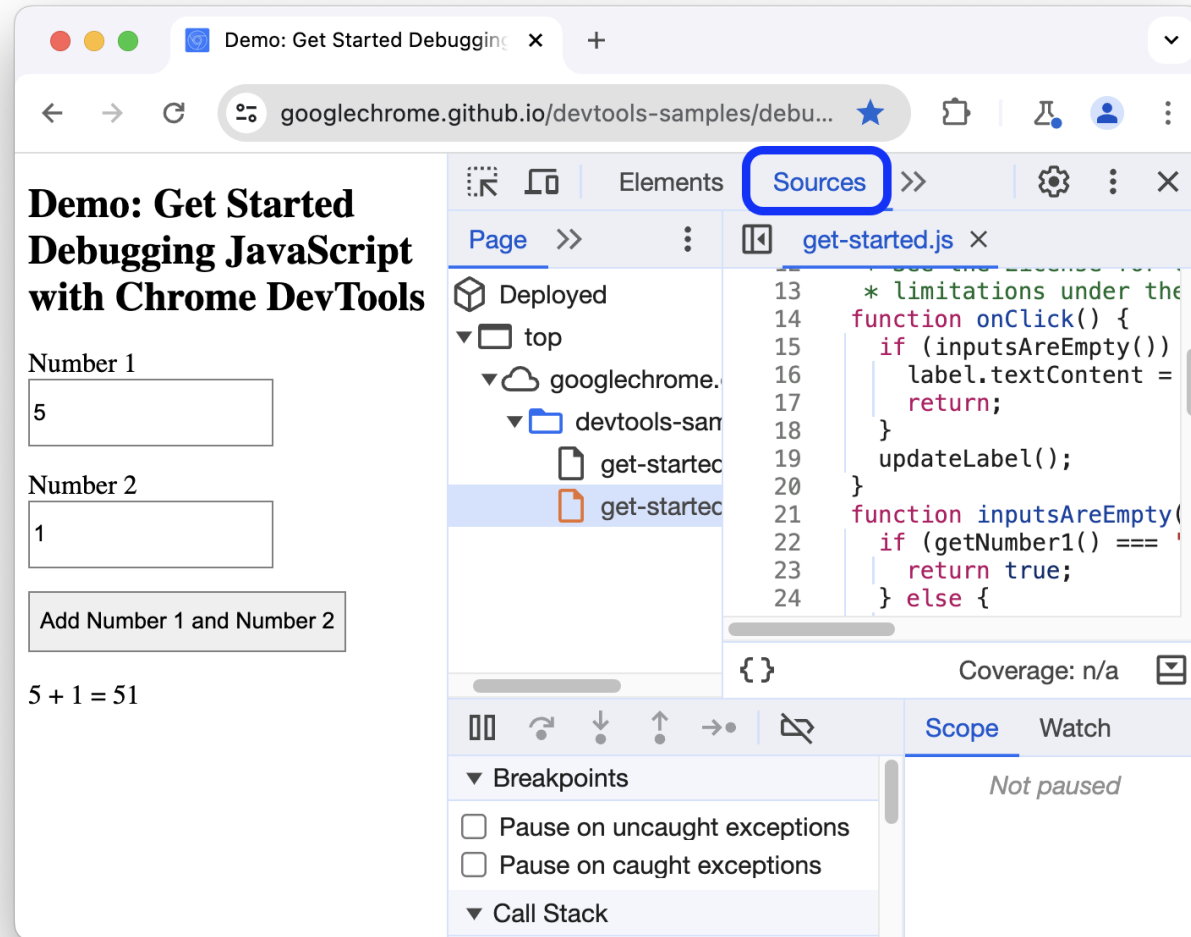
5. Dynamic body

# Debug JavaScript

---

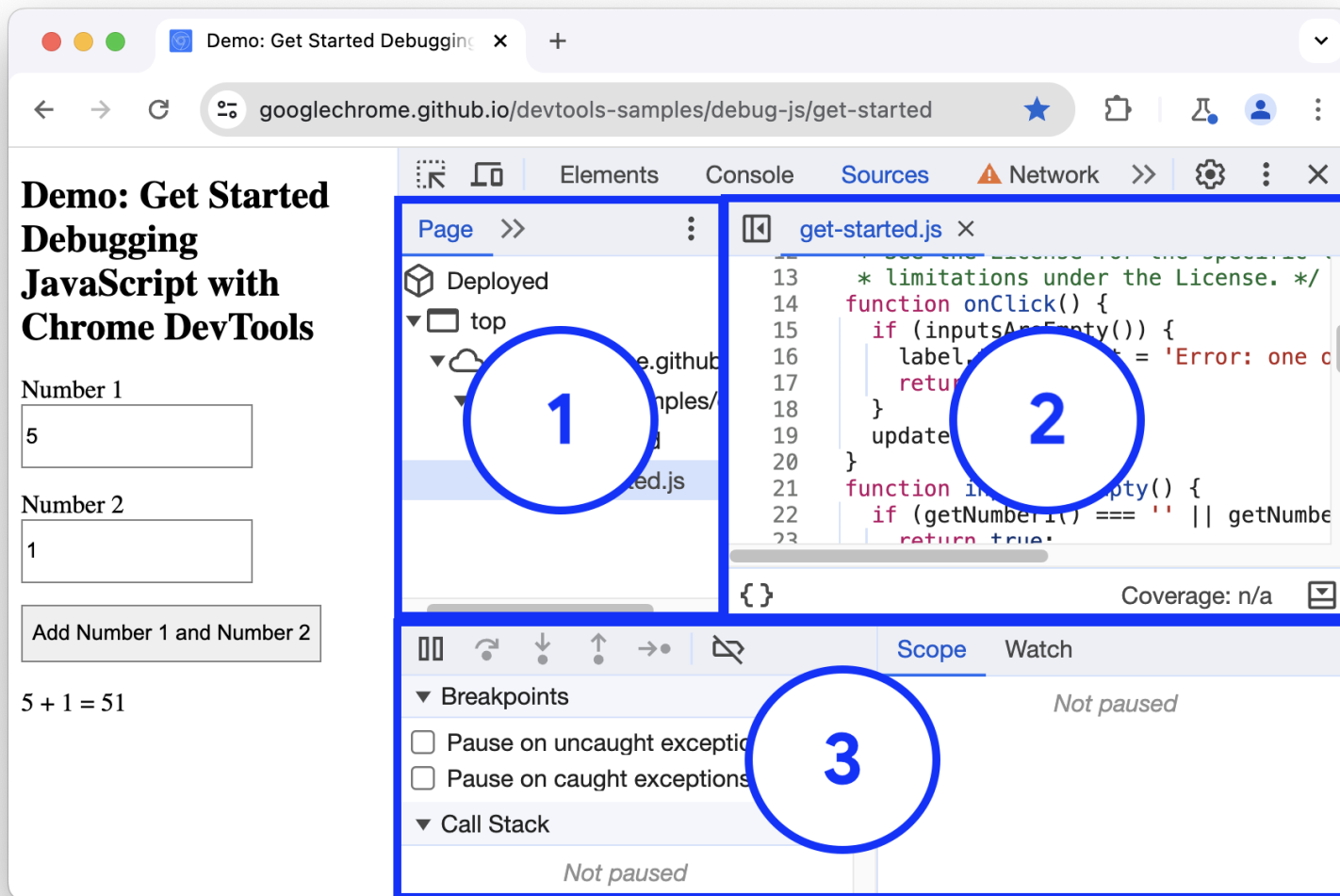
עבוד על תרגיל המחשבון

# Sources panel UI



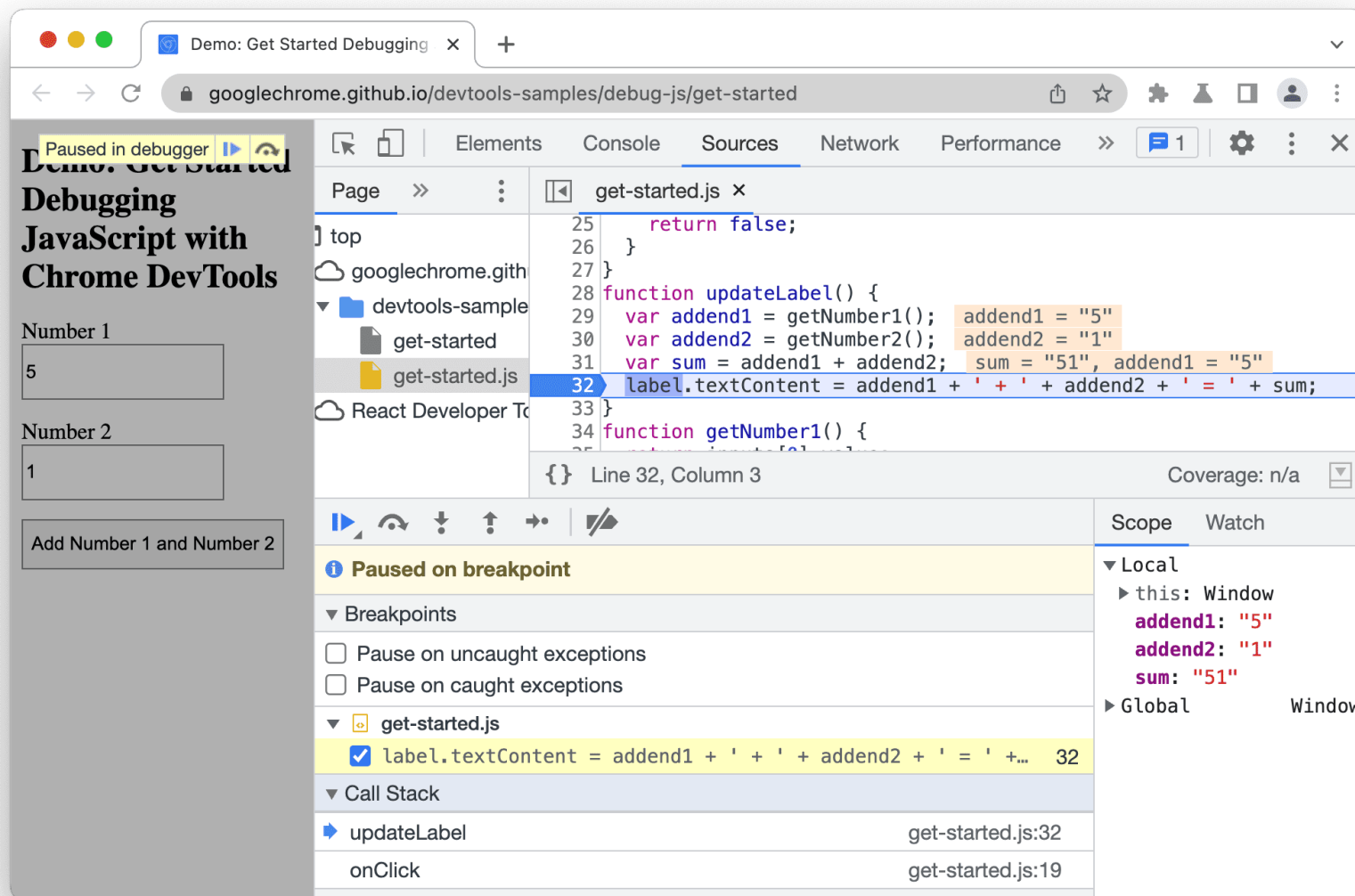
# Debug JavaScript

1. File tree
2. The opened file – code editor
3. Debugger section



# Debug JavaScript

1. Add breakpoint
2. Refresh the page
3. Step over, Step into
4. The scope window – show you local and global variables
5. Right click add watch



# if condition

---

```
let year = 2022;

if (year < 2022) {
    console.log('Too early...');
}
else if (year > 2022) {
    console.log('Too late');
}
else {
    console.log('Exactly!');
}
```

# ? : Condition

---

```
let result = condition ? value1 : value2;
```

```
let age = 18;
```

```
let accessAllowed = (age > 18) ? true : false
```

```
if (age < 3) {  
  message = 'Hi, baby!';  
} else if (age < 18) {  
  message = 'Hello!';  
} else if (age < 100) {  
  message = 'Greetings!';  
} else {  
  message = 'What an unusual age!';  
}
```



```
let message = (age < 3) ? 'Hi, baby!' :  
(age < 18) ? 'Hello!' :  
(age < 100) ? 'Greetings!' :  
'What an unusual age!';  
  
console.log(message);
```



# || (OR), && (AND), ! (NOT), ??

---

OR

```
let hour = 12;
let isWeekend = true;

if (hour < 10 || hour > 18 || isWeekend) {
  console.log('The office is closed.');// it is the weekend
}
```

AND

```
if (hour == 12 && minute == 30) {
  console.log('The time is 12:30');
}
```

NOT

```
console.log(!true); // false
console.log(!0); // true
```

a ?? b

---

**if a is defined, then a**

**if a isn't defined, then b**

```
// set height=100, if height is null or undefined  
height = height ?? 100;
```

# while

---

```
while (condition) {  
  // code  
  // so-called "loop body"  
}
```

```
let i = 0;  
while (i < 3) { // shows 0, then 1, then 2  
  console.log(i);  
  i++;  
}
```

# for

---

```
for (begin; condition; step) {  
    // "loop body"  
}
```

# skipping loop parts

---

```
let i = 0; // we have i already declared and assigned

for (; i < 3; i++) { // no need for "begin"
  console.log(i); // 0, 1, 2
}
```

```
let i = 0;

for (; i < 3;) {
  console.log(i++);
}
```

This makes the loop  
identical to while (i < 3).

## var vs. let

---

- var מאוד דומה ל let במרבית המקרים ניתן להחליף let ב var ולהפך ולצפות מהדברים לעבוד אותו דבר.
- בעבר היה רק var, כיום משתמשים ב let
- נעמוד על ההבדלים בין השניים

# “var” has no block scope

---

• ל var אין block scope

```
if (true) {  
    var test = true; // use "var" instead of "let"  
}  
console.log(test); // console.log: true
```

```
if (true) {  
    let test = true; // use "let"  
}  
  
console.log(test); // ReferenceError: test is not defined
```

# “var” has no block scope in loop

```
for (var i = 0; i < 10; i++) {  
    var one = 1;  
    // ...  
}
```

```
console.log(i);    // 10, "i" is visible after loop, it's a global variable  
console.log(one); // 1, "one" is visible after loop, it's a global variable
```

```
function sayHi() {  
    if (true) {  
        var phrase = "Hello";  
    }  
    console.log(phrase); // works  
}
```

```
sayHi();  
console.log(phrase); // ReferenceError: phrase is not defined
```

“var” has block scope in function



# “var” tolerates redeclarations

---

```
var user = "Pete";  
  
var user = "John"; // this "var" does nothing (already declared)  
// ...it doesn't trigger an error  
  
console.log(user); // John
```

```
let user;  
let user; // SyntaxError: 'user' has already been declared
```

– `let i = 0;`

```
for (i = 0; i < 3; i++) { // use an existing variable
  console.log(i); // 0, 1, 2
}
```

`console.log(i); // 3, visible, because declared outside of the loop`

```
for (let i = 0; i < 3; i++) {
  console.log(i); // 0, 1, 2
}
console.log(i); // error, no such variable
```

# switch

---

```
switch (x) {  
  case 'value1': // if (x === 'value1')  
    ...  
    [break]  
  
  case 'value2': // if (x === 'value2')  
    ...  
    [break]  
  
  default:  
    ...  
    [break]  
}
```

```
let a = 2 + 2;

switch (a) {
case 3:
console.log('Too small');
break;
case 4:
console.log('Exactly!');
break;
case 5:
console.log('Too big');
break;
default:
console.log("I don't know such values");
}
```

If there is **no break** then the **execution continues** with the next case **without any checks**.

```
let a = 2 + 2;
```

```
switch (a) {  
  case 3:  
    console.log('Too small');  
  case 4:  
    console.log('Exactly!');  
  case 5:  
    console.log('Too big');  
  default:  
    console.log("I don't know such values");  
}
```

```
console.log('Exactly!');  
console.log('Too big');  
console.log("I don't know such  
values");
```

# grouped two cases

---

```
let a = 3;
```

```
switch (a) {
```

```
case 4:
```

```
console.log('Right!');
```

```
break;
```

```
case 3: // (*) grouped two cases
```

```
case 5:
```

```
console.log('Wrong!');
```

```
console.log("Why don't you take a math class?");
```

```
break;
```

```
default:
```

```
console.log('The result is strange. Really.');
```

```
}
```

both 3 and 5 show the same message.

# תרגיל כיתה מסכם intro js

עליכם ליצור 4 תאים בשורה.

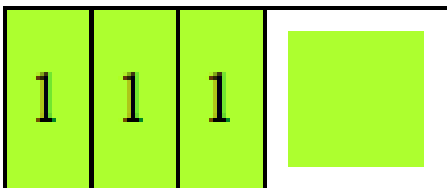
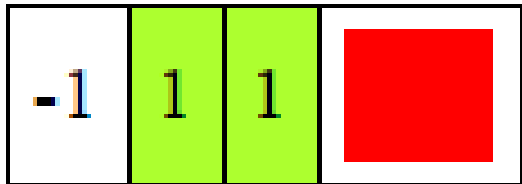
ב-3 תאים יופיע המספר -1.

לחיצה על תא עם מספר (-1), המספר הופך ל- (1) ומשתנה לצבע ירוק, לחיצה נוספת מחזירה לספרה (-1) והצבע נעלם.

בתא הרביעי יופיע div שצבעו תלוי בשורת התאים:

3 תאים ירוקים (ספרה 1) = div צבוע ירוק

כל מצב אחר = div בצבע אדום.



# שינוי ה-DOM

---

Javascript מספקת לנו את היכולת לשנות את תוכן הדף כרצוננו.  
הצורה הכי "פשוטה" לשינוי התוכן של אלמנט היא באמצעות תכונת `innerHTML`.  
התכונה מקבלת מחרוזת, אשר תוצג על המסך לפי קוד ה-HTML שרשום בה.

```
<body>
  <div id="myHtml"></div>

  <script>
    document.getElementById("myHtml").innerHTML = "<h1>Hello world</h1>";
  </script>
</body>
```



**Hello World**



# DHTML

---

באמצעות תכונת ה-innerHTML ניתן לדרוס את התוכן הקיים, להוסיף לקיים או למחוק את התוכן:

```
<body>
  <div id="myHtml"><h1>Hello World</h1></div>

  <script>
    //וספת תוכן לקיים
    document.getElementById("myHtml").innerHTML += "<h1>adding header</h1>";
    //ריסת הקיים
    document.getElementById("myHtml").innerHTML = "<h1>switching header</h1>";
    //חיקת התוכן
    document.getElementById("myHtml").innerHTML = "";
  </script>
</body>
```

# איך נבנה את הדף הבא בלחיצת כפתור?

This is a paragraph: 0



This is a paragraph: 1



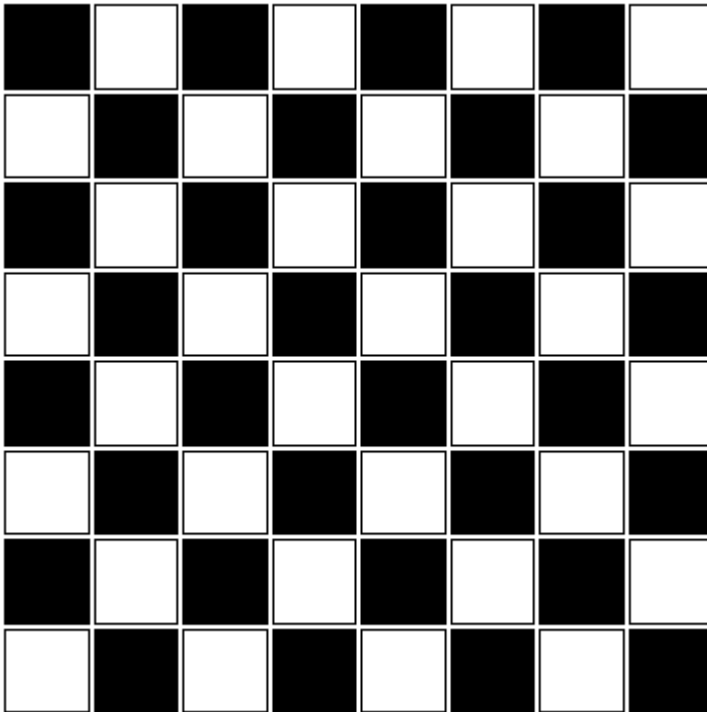
This is a paragraph: 2



```
function showImages() {  
    htmlStr = "";  
  
    for (var i = 0; i < 100; i++) {  
        htmlStr += "<div>";  
        htmlStr += "<p>This is a paragraph: " + i + "</p>";  
        htmlStr += '';  
        htmlStr += "</div>";  
    }  
    document.getElementById("ph").innerHTML = htmlStr;  
}
```

# תרגיל כיתה

---



שלב ראשון:

צור את לוח הדמקה הבא באופן דינמי

שלב שני:

לחיצה על תא כלשהו תשנה את צבעו לירוק

\*הפתרון שלי הוא באמצעות table מוזמנים

לעשות בדרך אחרת.

שלד לתרגיל כיתה דמקה:

```
<style>
    td {
        border: 1px solid #000;
        width: 30px;
        height: 30px;
    }

    .tdBlack {
        background-color: black;
    }

    .tdWhite {
        background-color: white;
    }
</style>
```

```
<body>
    <div id="ph"></div>
</body>
```

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

---

# פונקציות

# function decleration

---

• הצורה שכבר ראינו:

```
function sayHi()  
{  
  
  console.log("Hello");  
}
```

# function parameters

---

```
function functionName(parameter1, parameter2, parameter3)
{
  // code to be executed
}
```

```
function myFunction(a, b) {
  return a * b;
}

let x = myFunction(4, 3);
```

# function expression

---

```
let sayHi = function () {  
    console.log("Hello");  
};
```

- כאן אנו רואים משתנה בשם sayHi שהערך שלו הוא פונקציה.
- שימו לב: אין שם לפונקציה, אנו שמים את הפונקציה כערך של משתנה.
- אם נרצה לקרוא לפונקציה נקרא sayHi() עם סוגריים.
- אם נדפיס את המשתנה sayHi נקבל את הטקסט של הפונקציה.



# copy to another variable

---

```
function sayHi() { // (1) create
  console.log("Hello");
}
```

```
let func = sayHi; // (2) copy
```

```
func(); // Hello // (3) run the copy (it works)!
sayHi(); // Hello // this still works too (why wouldn't it)
```

```
let sayHi = function () { // (1) create
  console.log("Hello");
};
```

```
let func = sayHi;
// ...
```

# function declaration

---

בצורה הזו שהפונקציה מוגדרת – ניתן לקרוא לפונקציה לפני ההגדרה שלה.

```
sayHi("John"); // Hello, John
```

```
function sayHi(name) {  
    console.log(`Hello,  
${name}`);  
}
```

# function expression

---

בצורה זו שהפונקציה היא משתנה, ניתן לקרוא לה רק לאחר הגדרת המשתנה.

```
sayHi("John"); // error! ❌  
  
let sayHi = function (name) { //  
  (*) no magic any more  
    console.log(`Hello, ${name}`);  
};
```

# arrow function syntax

---

1. (param1, paramN) => expression
2. (param) => expression
3. () => expression
- 4.
5. (param1, paramN) => { statements }
6. param => { statements }

# 1. (param1, paramN) => expression

---

```
let sum = (a, b) => a + b;  
console.log(sum(1, 2)); // 3
```

/\* This arrow function is a shorter form of:

```
let sum = function(a, b) {  
  return a + b;  
};  
*/
```

## 2. param => expression

---

- אם קיים פרמטר אחד ניתן להוריד את הסוגריים ולכתוב כך:

```
let double = n => n * 2;  
console.log(double(3)); // 6
```

### 3. () => expression

---

• אם אין פרמטרים אפשר לכתוב סוגריים ריקים

```
let sayHi = () => console.log("Hello!");
```

```
sayHi();
```

## 4. param => { statements }

---

```
let func = (arg1, arg2, ..., argN) => expression;
```



```
let func = function (arg1, arg2, ..., argN) {  
    return expression;  
};
```



# array

---

אוסף אלמנטים מסודר האיבר הראשון במיקום 0

הצהרה:

```
let arr = [];
```

Or

```
let arr = new Array();
```

הצהרה והשמה:

```
let fruits = ["Apple", "Orange", "Plum"];
```

```
let fruits = ["Apple", "Orange", "Plum"];  
console.log(fruits[0]); // Apple  
console.log(fruits[1]); // Orange  
console.log(fruits[2]); // Plum
```

### Replace an element:

```
fruits[2] = 'Pear'; // now ["Apple", "Orange", "Pear"]
```

### Add a new one

```
fruits[3] = 'Lemon'; // now ["Apple", "Orange", "Pear", "Lemon"]
```

### Array length

```
let fruits = ["Apple", "Orange", "Plum"];  
console.log(fruits.length); // 3
```

### Show the whole array

```
let fruits = ["Apple", "Orange", "Plum"];  
console.log(fruits); // Apple,Orange,Plum
```

*An array can store elements of any type.*

```
// mix of values
let arr = ['Apple', 1, true, function () {
  console.log('hello'); }];

// get the number at index 1 and then show its name
console.log(arr[1]); // 1

// get the function at index 3 and run it
arr[3](); // hello
```

## Access last element

```
let fruits = ["Apple", "Orange", "Plum"];
```

```
console.log(fruits[fruits.length - 1]); // Plum
```

## Shorter syntax

```
let fruits = ["Apple", "Orange", "Plum"];
```

```
// same as fruits[fruits.length-1]
```

```
console.log(fruits.at(-1)); // Plum
```

# array methods

---

## **pop** •

- מוציא את האיבר האחרון מהמערך ומחזיר אותו

## **push** •

- מוסיף את האיבר הנתון לסוף המערך

## **shift** •

- מוציא את האיבר הראשון מהמערך ומחזיר אותו

## **unshift** •

- מוסיף את האיבר הנתון לתחילת המערך

## push, pop

---

```
let fruits = ["Apple", "Orange", "Pear"];  
console.log(fruits.pop()); // remove "Pear" and console.log  
return it  
console.log(fruits); // Apple, Orange
```

```
let fruits = ["Apple", "Orange"];  
fruits.push("Pear"); //is equal to fruits[fruits.length] =  
.....  
console.log(fruits); // Apple, Orange, Pear
```

# shift, unshift

---

```
let fruits = ["Apple", "Orange", "Pear"];  
console.log(fruits.shift()); // remove Apple and console.log  
it  
console.log(fruits); // Orange, Pear
```

```
let fruits = ["Orange", "Pear"];  
fruits.unshift('Apple');  
console.log(fruits); // Apple, Orange, Pear
```



# array methods

---

```
var arr1 = [1, "hello", "hi"];

arr1.unshift("11");    //add element to the beginning of the array
arr1.push("11");        //add element to the end of the array

arr1.shift();           //remove the first element from the array
arr1.pop();             //remove the last element from the array

arr1.length;           //the number of elements in the array

arr1[0] = "change";     //set value of the first element in the array or add it if not exists
```

איך ייראה המערך לאחר כל הפעולות?

*["change", "hello", "hi"]*

# loop over array

---

## Older way “for”

```
let arr = ["Apple", "Orange", "Pear"];
for (let i = 0; i < arr.length; i++) {
  console.log(arr[i]);
}
```

## another form of loop, “for..of”

```
let fruits = ["Apple", "Orange", "Plum"];
// iterates over array elements
for (let fruit of fruits) {
  console.log(fruit);
}
```

# loop over array

---

another form of loop, “for..in”

```
let fruits = ["Apple", "Orange", "Plum"];
for (let i in fruits) {
    console.log(i); // 0, 1, 2
    console.log(fruits[i]); // "Apple", "Orange", "Plum"
}
```

# array length

---

```
let fruits = [];  
fruits[123] = "Apple";
```

```
console.log(fruits.length); // 124
```

---

```
let arr = [1, 2, 3, 4, 5];
```

```
arr.length = 2; // truncate to 2 elements  
console.log(arr); // [1, 2]
```

```
arr.length = 5; // return length back  
console.log(arr[3]); // undefined: the values do not return
```

# תרגיל ביתה

---

## פלט:

"row 0"

" 1"

" 2"

" 1"

" 24"

"row 1"

...

...

...

כתוב פונקציה המקבלת את מערך הדו מימדי הבא של מספרים

```
var a = [[1, 2, 1, 24],  
          [8, 11, 9, 4],  
          [7, 0, 7, 27],  
          [7, 4, 28, 14],  
          [3, 10, 26, 7]];
```

ומשתמשת בלולאה מקוננת על מנת להדפיס את אברי מערך לפי שורות

# תרגיל ביתה

---

- כתוב פונקציה שמקבלת מערך חד מימדי של מספרים ומחרוזות ומדפיסה את האבר שמופיע הכי הרבה פעמים וכמה פעמים הוא מופיע .

• קלט:

•

```
var arr1 = [3, 'a', 'a', 'a', 2, 3, 'a', 3, 'a', 2, 4, 9, 3];
```

•

פלט:

- a ( 5 times )

# Array forEach

---

```
const array1 = ['a', 'b', 'c'];
```

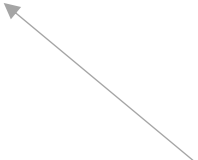
```
array1.forEach((element) => console.log(element));
```

```
// Expected output: "a"
```

```
// Expected output: "b"
```

```
// Expected output: "c"
```

מבצע את הפעולה על כל אחד מהאלמנטים



# Converting a for loop to forEach

---

```
let items = ["item1", "item2", "item3"];  
let copyItems = [];
```

// before

```
for (let i = 0; i < items.length; i++) {  
  copyItems.push(items[i]);  
}
```

// after

```
items.forEach((item) => {  
  copyItems.push(item);  
});
```



# תרגיל ביתה

---

• פתח את ה skelton בקובץ jewishManDOMClassEx.zip הרץ אותו הוסף את האירועים הרשומים בדף מועתק גם כאן לנוחיותכם:

1. **onclick** on the **button** - Copy the text to the right div
2. **onmouseover** on the **smiley** - Add the image to the right div
3. **onclick** on the **jewish** smiley – change all the images in the right div to the jewish smiley
4. **ondblclick** on the **eraser** - remove all the content from the right div

---

Copy Text



1. onclick on the button - Copy the text to the right div
2. onmouseover on the smiley - Add the image to the right div
3. onclick on the jewish smiley - change all the images in the right div to the jewish smiley
4. ondblclick on the eraser - remove all the content from the right div