

משתנים

JAVASCRIPT

מה נלמד?

- מהם משתנים ואיך יוצרים אותם

- String interpolation

- Operators

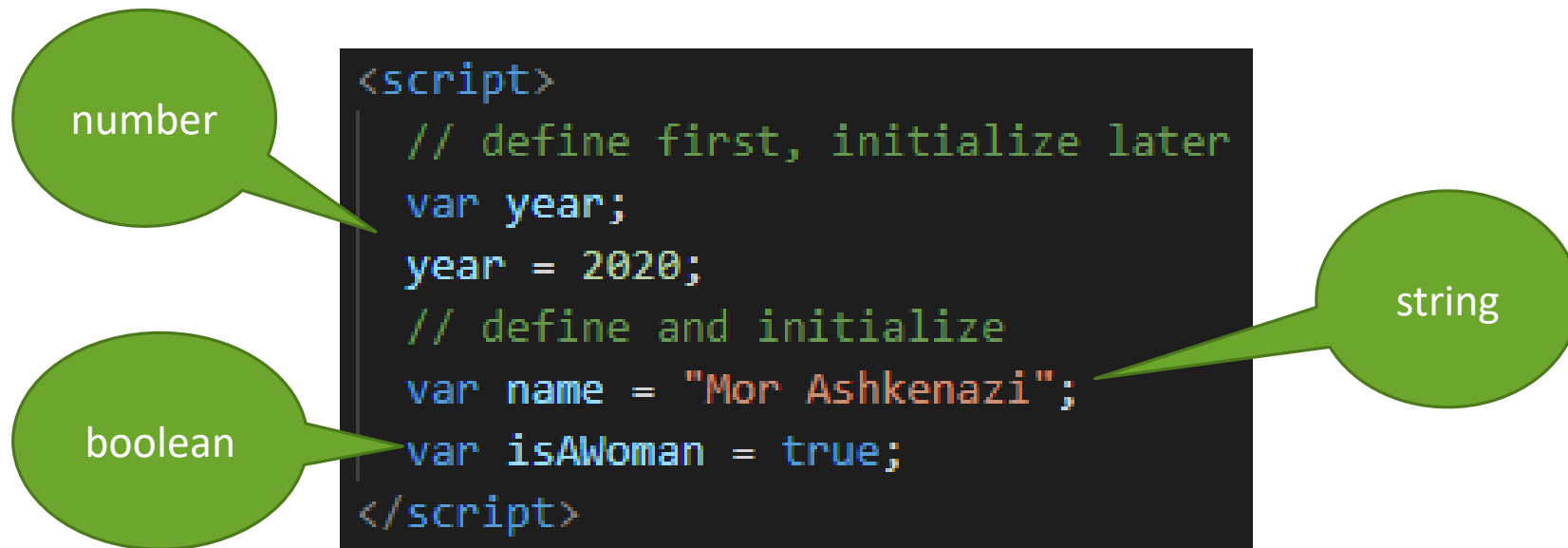
- פרמטרים לפונקציה ו-scope

- Hoisting

- Variables in es6

Variables

- משתנים הם מעין "קופסאות" או "מיכלים" בשפות תכנות שונות אך במקום להכיל דברים מוחשיים הם מכילים מידע.
- לדוגמה:



Variables

חוקי שמות משתנים:

- שם משתנה חייב להתחיל באות באנגלית או קו תחתי
- יתר התווים במשתנה חייבים להיות בשפה האנגלית, מספרים או קו תחתי בלבד
- הכרזה על משתנה תתבצע באמצעות המילים השמורות: `var`, `let`, `const`
- משתנה עם אות גדולה בהתחלה ומשתנה עם אות קטנה בהתחלה - הם שני משתנים שונים

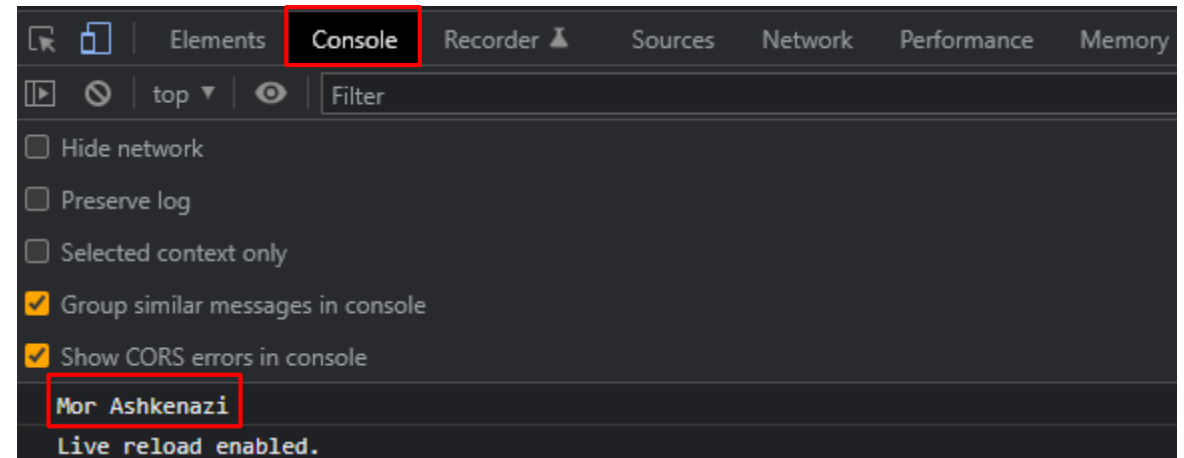
```
var name = "Mor";
```

```
var Name = "Mor";
```

console.log()

פקודה המאפשרת הדפסה לקונסול שנמצא בדפדפן
מאפשרת לנו לראות תוכן של משתנים מבלי שהמשתמש יהיה חשוף לכך באופן ישיר

```
<script>  
  // define first, initialize later  
  var year;  
  year = 2020;  
  // define and initialize  
  var name = "Mor Ashkenazi";  
  var isAWoman = true;  
  // print to console  
  console.log(name);  
</script>
```



console.log()

דוגמאות נוספות:

```
// console.log examples  
console.log(5 * 2);  
console.log(year + 3);  
console.log(name + " Full Stack Developer");  
console.log("My age is", 18);
```

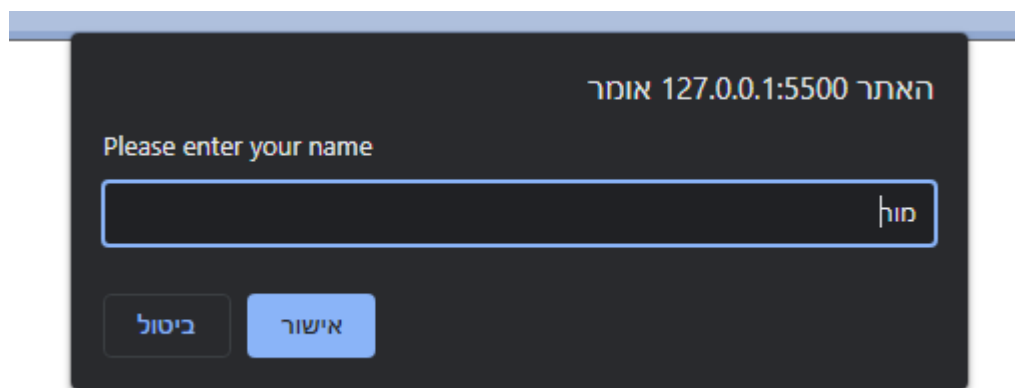


```
10  
2023  
Mor Ashkenazi Full Stack Developer  
My age is 18
```

prompt

מאפשר לקלוט מהמשתמש קלט

```
<h1 id="h1_id"></h1>
<script>
  var name = prompt("Please enter your name");
  document.getElementById("h1_id").innerHTML = name;
</script>
```



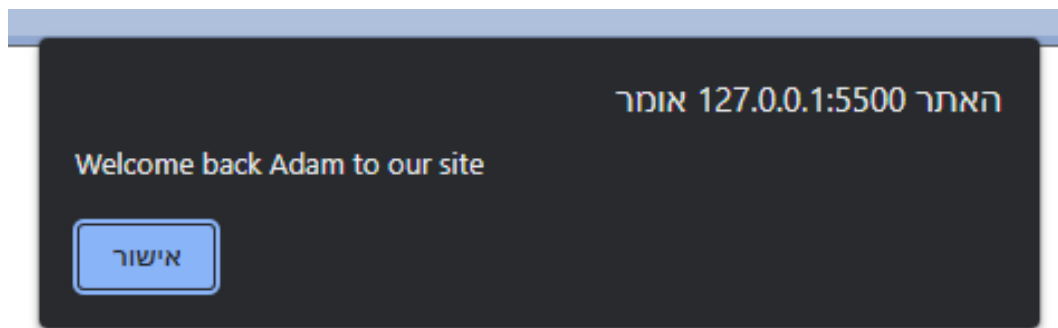
מור

String interpolation

שרשור משתנים למחרוזת – שיטה 1

■ שימוש באופרטור +

```
<script>
  var userName = "Adam";
  var message = "Welcome back " + userName + " to our site";
  alert(message);
</script>
```



String interpolation

שרשור משתנים למחרוזת – שיטה 2

שימוש בסימון `\${}`

```
<script>
  var userName = "Adam";
  var message = `Welcome back ${userName} to our site`;
  alert(message);
</script>
```

האתר 127.0.0.1:5500 אומר

Welcome back Adam to our site

אישור



operators

פקודות המאפשרות לבצע מניפולציה חשבונית על מספרים

סימון	משמעות
+	חיבור מתמטי או שרשור מחרוזות
-	חיסור מתמטי
*	כפל מתמטי
/	חילוק מתמטי
%	מודולו – בדיקת שארית חלוקה
**	חזקה מתמטית

operators

דוגמאות:

```
<script>
  var num1 = 2,
      num2 = 3;
  console.log(num1 + num2);
  console.log(num1 - num2);
  console.log(num1 * num2);
  console.log(num1 / num2);
  console.log(num1 % num2);
  console.log(num1 ** num2);
</script>
```



5
-1
6
0.6666666666666666
2
8

operators

פקודות המאפשרות לבצע מניפולציה חשבונית על מספרים

```
<script>
  var num1 = 2;
  num1 += 2; // num1 = 4
  num1 -= 2; // num1 = 2
  num1 *= 2; // num1 = 4
  num1 /= 2; // num1 = 2
  num1++; // num1 = 3
  num1--; // num1 = 2
</script>
```

סימון	דוגמה	שקול ל..
+=	num += 2	num = num + 2
-=	num -= 2	num = num - 2
*=	num *= 2	num = num * 2
/=	num /= 2	num = num / 2
++	num++	num = num + 1
--	num--	num = num - 1

פרמטרים לפונקציה

- פונקציה יכולה לקבל פרמטרים
- בכדי להדגים את השימוש, נניח שמחיר מוצר כלשהו, הוא 150 ₪ ויש עליו הנחה של 20%, נכתוב פונקציה שמחשבת את המחיר הסופי של המוצר לאחר הנחה:

```
<script>
function calcDis() {
  var result = 150 - (150 * 20) / 100;
  document.write(`<h1> The final price is ${result}`);
}
calcDis();
</script>
```

פונקציה
עם ערכים
ספציפיים

קריאה
לפונקציה

The final price is 120

```
<script>
function calcDis(price, discount) {
  var result = price - (price * discount) / 100;
  document.write(`<h1> The final price is ${result}`);
}
calcDis(150, 20);
</script>
```

פונקציה
שמקבלת
ערכים

קריאה
לפונקציה

The final price is 120

פרמטרים לפונקציה

- פונקציה יכולה להחזיר ערך

החזרת
הערך

```
<script>
function calcDis(price, discount) {
  var result = price - (price * discount) / 100;
  return result;
}
document.write(`<h1> The final price is ${calcDis(150, 20)}`);
</script>
```

קריאה
לפונקציה

Scope – global and local

מה יהיה ערך money בסיום ריצת התוכנית?

```
<script>
  var money = 100;
  function func1() {
    | money *= 2;
  }

  function func2() {
    | var money = 50;
  }

  function func3() {
    | money++;
  }

  func1();
  func2();
  func3();
</script>
```

Function hoisting

- האם הקוד תקין?

```
<script>
  myFunc();

  function myFunc() {
    console.log("Hello!");
  }
</script>
```

- התשובה היא: כן!
- ל-javascript תכונה מיוחדת – לפני הרצת שורות הקוד קודם כל נקראות הגדרות הפונקציות ולכן ניתן לזמן פונקציה לפני הגדרתה.

Variable hoisting

- מה תהיה ההדפסה בקונסול?

```
<script>  
  console.log(pi);  
  var pi = 3.14;  
</script>
```



```
<script>  
  var pi;  
  console.log(pi);  
  pi = 3.14;  
</script>
```

- התשובה היא: undefined
- ל-javascript תכונה מיוחדת – לפני הרצת שורות הקוד קודם כל נקראות הגדרות המשתנים var (לא אתחולם).

Variables in ES6

החל מגרסה ES6 של javascript נוספו עוד שתי אופציות להגדיר משתנים:

- let – מייצג בדרך כלל משתנים מקומיים
- const – מייצג בדרך כלל משתנים שערכם קבוע

דומה
בהצהרתו
ל-var

```
<script>  
  var name = "Sharon";  
  let age = 41;  
  const id = 202221111;  
  id = 5;  
</script>
```

✖ Uncaught TypeError: Assignment to constant variable.
at home.html:105:10

Variables in ES6

מה יהיה הפלט בקונסול?

1

```
<script>
  var x = 5;
  {
    let x = 10;
  }
  console.log(x);
</script>
```

2

```
<script>
  var x = 5;
  {
    var x = 10;
  }
  console.log(x);
</script>
```