# JavaScript – מתקדם

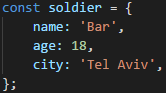
יש לפתוח את המסמך באפליקציה הדסקטופית.

Data Types שונים

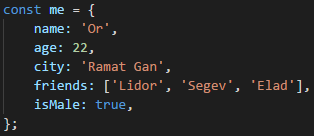
### עבודה עם אובייקטים

בJS כמעט כל דבר הוא אובייקט – מערכים, מחרוזות, סוגי נתונים, פונקציות ועוד. לעתים אנחנו נדרשים ליצור בעצמנו אובייקטים חדשים שלא הוגדרו כחלק מן השפה.

הדרך שנלמד ליצירת אובייקטים היא על ידי שימוש בObjects Literals. OL היא פשוט רשימה של שמות תכונות (המופרדים בפסיקים) וערכים לתכונות אלו (המופרדים בנקודתיים) המגדירים יחדיו את האובייקט, תכונותיו וערכי התכונות הללו.



ניתן להגדיר לאובייקטים שלנו תכונות שונות כגון: מחרוזת, מספר, בוליאני, ואפילו מערכים שונים:



אחרי שיצרנו את האובייקט, נוכל לגשת אליו כמו לכל אובייקט שאנחנו מכירים מ Java:



העשרה: שרשור אלמנטים ב JavaScript יכול להיעשות גם כך:



### מתודות של אובייקטים

ניתן להגדיר מתודה גם כתכונה של אובייקט. נגדיר מתודה לאובייקט כפי שנגדיר כל פונקציה בהבדל אחד, שפונקציה זו תוגדר כתכונה של אובייקט. הצורה הכללית להגדרת מתודה תהיה:

const me = {

name : "Yosi",

age : 22,

printMe: function() {

return `my name is ${this.name}`

}

}

ניתן לקצר את ההגדרה של הפונקציה, ולרשום בעזרת הסינטקס הבא (שימו לב לprintMe)

const me = {

name : "Yosi",

age : 22,

printMe() {

return `my name is ${this.name}`

}

}

וכדי להפעיל את המתודה פשוט ניגש אליה כאילו היא מאפיין של האובייקט (בתוספת סוגריים): 

הסבר מורחב על אובייקטים:

יש לקרוא בקישור הבא:

[לחץ כאן](https://javascript.info/object)

עד הכותרת Square brackets כולל

עבודה עם מספרים:

יש לקרוא בקישור הבא:

[לחץ כאן](https://javascript.info/number)

עד הכותרת Rounding כולל

על עקרון הe אפשר רק לרפרף ולהבין מה כתוב שם

עבודה עם מחרוזות:

יש לקרוא בקישור הבא:

[לחץ כאן](https://javascript.info/string)

עד הכותרת Getting a substring כולל

אפשר לדלג על תת הכותרת " The bitwise NOT trick"

עבודה עם מערכים:

יש לקרוא בקישור הבא:

[לחץ כאן](https://javascript.info/array)

עד הכותרת new Array() כולל

עבודה עם פונקציות:

יש לקרוא בקישור הבא:

[לחץ כאן](https://javascript.info/function-basics)

עד הכותרת Default values כולל

עבודה עם תאריכים:

יש לקרוא בקישור הבא:

[לחץ כאן](https://javascript.info/date)

עד הכותרת Setting date components כולל

לרוץ מהר על המידע ורק להבין את העיקרון

שגיאות:

יש לקרוא בקישור הבא:

[לחץ כאן](https://javascript.info/try-catch)

לדלג על הכותרת Throwing our own errors, על Rethrowing ועל Global catch

הדגמה של try, catch, finally:

[לחץ כאן](https://codepen.io/codebeast/pen/qVamZm)

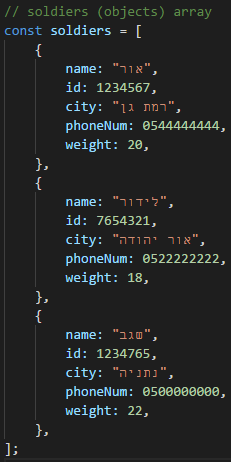
תסתכלו על הקוד בJS בצד ימין

פעולות על מערכים

הפעולות המוכנות ששפת JavaScript מציעה על מערכים נחשבות לכלי חזק בשפה. במהלך הפיתוח שלנו ב JS נשתמש המון בפעולות אלה כי הן יאפשרו לנו לבצע תהליכים וחישובים מורכבים בשורה אחת בלבד ובאופן קריא וברור.

שימו לב: כל אחת מהפעולות הללו דורשת לקבל חישוב (באמצעות פונקציה) אשר בדר"כ לא יהיה מורכב כשלעצמו. זו תהיה הזדמנות נהדרת להשתמש ב arrow functions, תוכלו להשתמש בfunction רגילה או בפונקציה אנונימית (כותרת הבאה).

כאמור במהלך הל.ע נכיר מניפולציות וחישובים על מערכים, את הדוגמאות לכל אחד מהם נבצע על מערך לדוגמה של אובייקטים מסוג "חייל" שנקרא לו soldiers ונראה כך:



פונקציות אנונימיות –

פוקנציה ללא שם שמשמשת להכנסתה לתור מקרים ספציפיים. דוגמאות:

1. הכנסת פונקציה אנונימית לתוך משתנה (שהוא כרגע הפונקציה)
2. let myFunc = function() {
3. alert('Hello World');
4. }

2. הכנסת פונקציה אנונימית לתוך פונקציות על מערכים: יוזכר בהמשך

* Map
* הגדרה: הפונקציה map() יוצרת מערך חדש שאבריו הם תוצאת חישוב של פונקציה המסופקת לו על כל אחד מאיברי המערך המקורי.

incrementByOne = function (element) {

return element + 1;

}

myArray = [1,2,3,4];

myArray.map(incrementByOne); *// returns [2,3,4,5]*

*או*

myArray.map(function (element) {

return element + 1;

}) *// returns [2,3,4,5]*

*// anonymous function (דוגמא 2 בפונקציות אנונימיות)*

* סינטקס: const newArray = array.map(function);

שימו לב: הפונקציה ש map מצפה לקבל, תקבל בכל פעם פרמטר (איבר יחיד מהמערך) ותחזיר תוצאת חישוב כלשהי המבוססת על אותו איבר (שיהווה איבר במערך החדש).

* מתי נשתמש: נשתמש ב map() בדרך כלל כדי ליצור טרנספורמציה על מערך קיים, כלומר לשנות את מבנה הערכים/האובייקטים שהוא מכיל.
* דוגמה: יש לנו את מערך החיילים soldiers ובשביל סקריפט שמוסיף חיילים לקבוצה בוואטסאפ באופן אוטומטי אנו זקוקים למערך של מספרי טלפון בלבד. הפונקציה map תאפשר לנו ליצור מערך כזה בשורה אחת בלבד:

soldiers.map(function (soldier) {

return soldier.phonelNumber;

}); *// returns [0544444444, 0522222222, 0500000000]*

* Find
* הגדרה: הפונקציה find() מחזירה ערך יחיד מתוך מערך אשר מתאים לתנאי (פונקציה בוליאנית) שמוגדר לה.

שימו לב: במידה וכמה איברים במערך יתאימו לתנאי – רק הראשון יוחזר.

* סינטקס: const value = array.find(function);

שימו לב: הפונקציה ש find מצפה לקבל, תקבל בכל פעם (עד תנאי העצירה) פרמטר (איבר יחיד מהמערך) ותבצע עליו איזשהו חישוב בעל תוצאה בוליאנית אותו תחזיר.

* מתי נשתמש: נשתמש ב find() בדרך כלל כדי "לשלוף" איבר מתוך מערך בהתאם לתנאי ספציפי.
* דוגמה: יש לנו את מערך החיילים soldiers ובשביל להדפיס פרטי חייל על תעודת ההצטיינות אנו זקוקים לחייל (אובייקט יחיד) שמספרו האישי הוא 1234567. הפונקציה find תאפשר לנו למצוא (לשלוף) את האובייקט הנ"ל בשורה אחת בלבד:

soldiers.find(function (soldier) {

return soldier.id === 1234567;

}); *// מוחזר האובייקט של אור מתוך המערך*

* Filter
* הגדרה: הפונקציה filter() יוצרת מערך חדש אשר מכיל את כל איברי המערך המקורי אשר עוברים תנאי (פונקציה בוליאנית) שמוגדר לה.
* סינטקס: const newArray = array.filter(function);

שימו לב: הפונקציה ש filter מצפה לקבל, תקבל בכל פעם פרמטר (איבר יחיד מהמערך) ותבצע עליו איזשהו חישוב בעל תוצאה בוליאנית אותו תחזיר.

* מתי נשתמש: נשתמש ב filter() בדרך כלל כדי "לסנן" איברים מתוך מערך בהתאם לתנאי ספציפי.
* דוגמה: יש לנו את מערך החיילים soldiers וכדי לדעת אילו חיילים חוזרים באוטובוס לר"ג לאחר סיור באור יהודה אנו זקוקים למערך חיילים שאינו מכיל את אלו שגרים באור יהודה. הפונקציה filter תאפשר לנו "לסנן" את האובייקטים הלא רצויים וליצור בהתאם מערך חדש בשורה אחת בלבד:

soldiers.filter (function (soldier) {

return soldier.city != "אור יהודה";

}); *// מוחזר האובייקט של אור ושל שגב מתוך המערך*

* Reduce
* הגדרה: הפונקציה reduce() מחזירה ערך יחיד אשר מהווה תוצאת חישוב של פונקציה המסופקת לה שמסתמכת על כלל האיברים במערך המקורי.
* סינטקס: const value = array.reduce(function);

שימו לב: הפונקציה ש reduce מצפה לקבל, תקבל בכל פעם שני פרמטרים ותבצע בעזרתם איזשהו חישוב שאת ערכו היא תצבור. הפרמטר הראשון הוא "הסוכם" אשר ערכו ההתחלתי הוא 0 ובכל קריאה חוזרת לפונקציה ערכו יהיה כערך ההחזר של הקריאה הקודמת. הפרמטר השני יהיה ערכו של התא הבא במערך.

* מתי נשתמש: נשתמש ב reduce() בדרך כלל כדי "לצבור"/לסכום איברים (או ערכים מתוך איברים) מתוך מערך בהתאם לחישוב ספציפי (חיבור, חילוק וכו').
* דוגמה: יש לנו את מערך החיילים הידוע soldiers ובשביל חישוב כלשהו (לדוגמה כדי לדעת האם כלל החיילים יכולים להיכנס למעלית עם מגבלת משקל) אנו זקוקים לערך המהווה את משקל כלל החיילים במערך. הפונקציה reduce תאפשר לנו לסכום את כלל הערכים בשורה אחת בלבד:

soldiers.map(function (soldier) {

return soldier.weight;

}).reduce(function (soldiersWeightA, soldiersWeightB) {

return soldiersWeightA + soldiersWeightB;

}); *// return 60*

שימו לב: בדוגמה לעיל בצענו את הסכימה בשני שלבים:

1. בצענו map כדי ליצור מערך של ערכי המשקל של כל החייל ([20, 18, 22]), מערך של ערכים מסוג integer שיהיה קל יותר לסכום.
2. בצענו על מערך ערכי המשקל את הסכימה בעזרת reduce.

Best practices בJS –

חובה לקרוא [כאן](https://www.w3schools.com/js/js_best_practices.asp)

תרגול פעולות על מערכים

* 1. תמונה שמכילה טקסט

     התיאור נוצר באופן אוטומטי
  2. תמונה שמכילה טקסט

     התיאור נוצר באופן אוטומטי
  3. תמונה שמכילה טקסט

     התיאור נוצר באופן אוטומטי
  4. בהינתן מערך של מילים, תכתבו פונקציה שמחזירה מערך של כל המילים שאורכן עולה על 6 תווים.
  5. בהינתן מערך של אובייקטים, מצא את האובייקט הראשון שהid שלו גדול מ13.

1. Set the next object in your browser like this :

const a = {  
  num: 0  
}

1. When we are doing a.valueOf() what should we get?

Make it return the num’s value.

[If you’re a bit stuck, there is a nice explanation here](https://www.google.co.il/search?q=how+to+search+by+myself+in+google&source=lnms&sa=X&ved=0ahUKEwjnrafMw-TYAhXOIlAKHbOEBxwQ_AUICSgA&biw=1237&bih=660&dpr=1.5)

1. Why do the next 2 commands don't do the same thing?

a.valueOf = function() {  
  return this.num;  
}

a.valueOf = () => this.num;

7.

If(2)

    console.log(“YAY”);

// the output is - YAY

So we can understand that 2 is true, not only 1, and of course 0 is false,

BUT

Why does *this* happen?

console.log(2==true);

// false

Hint - <http://www.ecma-international.org/ecma-262/#sec-abstract-equality-comparison>

בונוס:

תמונה שמכילה טקסט

התיאור נוצר באופן אוטומטי

ועוד משהו קטן לפני שמסיימים:

