

# ללמוד ג'אווהסקורייפט בנברית

רן בר-זיך



**Wix**Engineering

**Outbrain**  
Engineering

**R** Really Good

**Chegg**®

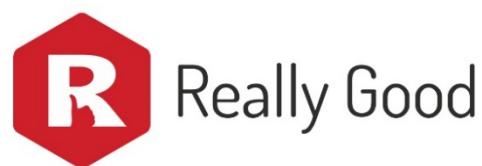
**Ono**

הקריה האקדמית אונ  
Ono Academic College  
החוג למדעי המחשב

# לימוד ג'אווהסקריפט בעברית

## רן בר-זיך

מהדורה: 2.0.0



כל הזכויות שמורות © רן בר-זיך, 2019.

ספר זה הוא יצירה המוגנת בזכויות יוצרים. אתה קיבלת רישיון לא-בלודי, לא-ឈודוי, אישי, בלתי ניתן להעbara (למעט על פי דין), ובلتוי ניתן להסבה לעשות שימוש אישי בספר זה לצרכים לימודים בלבד.

אסור לך להעתיק את הספר, לשכפל אותו, לצורך יצירות נגזרות ממנו או לפרסם אותו בכל צורה אחרת.

מותר לך לצלט קטעים קצרים מהספר במסגרת הגנת שימוש הוגן, בלויר פסקה או שתים, כאשר אתה מפנה למקור ומצכיר את רן בר-זיק כמחבר הספר.

הדוגמאות המובאות בספר זה הן בבעלות של רן בר-זיק, ואסור לך להשתמש בהן בתוך תוכנות שיתפתח. אם אתה רוצה להכניס אותן לפרויקט שלך, שלח מייל ונדבר על זה.

עריבכה לשונית: יעל ניר  
הגאה: חנן קפלן  
עיצוב הספר והכricaה: טל סולומון ורדי ([tsv.co.il](http://tsv.co.il))

הפקה: בריכה – סוכנות לסופרים  
[www.kricha.co.il](http://www.kricha.co.il)



## תוכן העניינים

<b>11</b>	<b>על הספר</b>
<b>12</b>	<b>על המחבר</b>
<b>13</b>	<b>על העורכים הטכניים</b>
<b>13</b>	<b>דניאל שטרנלייבט</b>
<b>13</b>	<b>gil finik</b>
<b>13</b>	<b>יגאל סטקלוב</b>
<b>14</b>	<b>תום ביגליין</b>
<b>14</b>	<b>צחי נמנី</b>
<b>15</b>	<b>על ג'אווהסקריפט</b>
<b>17</b>	<b>אין לומדים</b>
<b>17</b>	ארגנו לעצמכם סביבת עבודה מסודרת
<b>17</b>	קראו את הפרקים לפי הסדר
<b>17</b>	קראו כל פרק פעמיים וכותבו לעצמכם את כל הדוגמאות
<b>18</b>	פתרו את התרגילים לדוגמה בסוף כל פרק
<b>18</b>	התיעצו עם אחרים
<b>18</b>	הגיעו לسدנאות ולミיטאפים
<b>18</b>	תרגלו, תרגלו, תרגלו
<b>19</b>	<b>התקנת סביבת העבודה ודרכו הלימוד</b>
<b>28</b>	<b>משתנים</b>
<b>33</b>	<b>טקסט</b>
<b>37</b>	<b>מספרים</b>
<b>39</b>	מציאת השאריות
<b>41</b>	אופרטורים מקוצרים
<b>47</b>	<b>סוגי מידע פרימיטיביים נוספים</b>
<b>47</b>	בוליאני
<b>47</b>	משתנה לא מוגדר
<b>48</b>	רייך

48	Symbol
49	מציאת הסוג של המשתנה
53	הערות
56	בקרת זרימה – משפטי תנאי
59	אופרטורים השוואתיים נוספים
63	אופרטורים לוגיים
66	אופרטור שלילי
68	אופרטור תנאי
69	אופרטורים המשווים ערכים
79	switch case
88	קבועים
91	בקרת זרימה – פונקציות
95	פונקציה עם ארגומנטים
98	ארוגומנטים עם ערכים דיפוליטיים
100	הפונקציה כאובייקט
100	Hoisting
102	closure
105	פונקציה אוניבימית ופונקציית חץ
106	פונקציה אוניבימית במשתנה
107	פונקציה אוניבימית שմבזלת מהסקופ הגלובלי
116	אובייקטים
122	מחיקת מפתח
122	הכנסת פונקציה כערך
123	אובייקט בקבוע
130	מערכות
135	מערכות ומחזרות טקסט
138	this-new
146	תבנית טקסט
151	lolæot
151	lolæot for
161	lolæo אינסופית

161.....	לולאת <code>while</code>
163.....	לולאת <code>do while</code>
163.....	לולאת <code>forEach</code>
166.....	לולאת <code>for of</code>
171.....	לולאת <code>map</code>
174.....	לולאת <code>filter</code>
178.....	לולאת <code>sort</code>
178.....	לולאות על אובייקטים
178.....	לולאות <code>for in</code>
181.....	<code>Object.keys</code>
190 .....	<b>ג'אווהסקרייפט בסביבת דפדף</b>
190.....	הסבר כללי על HTML
193.....	מזהים של תגיות HTML
193.....	גישה אל תגיות באמצעות ג'אווהסקרייפט
194.....	אלמנטים של HTML מתרגמים לאובייקטים של ג'אווהסקרייפט
196.....	אירועים עם HTML וג'אווהסקרייפט
198.....	הצמדת אלמנטים של HTML לאלמנטים אחרים של HTML באמצעות ג'אווהסקרייפט
200.....	שינויי עיצוב
201.....	סלקטוריים של DOM
206.....	אירועים נוספים
208.....	פעוף של אירועים
211.....	הצמדת אירועי ג'אווהסקרייפט לאלמנטים ב-HTML
221 .....	<b>דיבאגינג</b>
228 .....	<b>אובייקטים גלובליים ואובייקטים מובנים</b>
231.....	<code>parseInt</code>
231.....	<code>eval</code>
232.....	<code>Math</code>
233.....	<code>Date</code>
236.....	<code>JSON</code>
237.....	<code>setTimeout</code>
244 .....	<b>ביטויים ורגולריים</b>

252	טיפול בשגיאות
254	finally
258	כבני נתונים מסוג <i>Set</i> - <i>Map</i>
258	<i>Map</i>
259	<i>Set</i>
264	תכנות אסינכריוני – קולבקים
274	<i>Promises</i>
278	שרשור הבטחות
281	קיבוץ הבטחות
284	פונקציית <code>async</code>
291	<b>AJAX</b>
293	מתודות של HTTP וארגוניים נוספים
297	<b>ES6 Classes</b>
304	ומה עבשינו?
306	<b>Best Practices</b>
306	מה זה <b>Best Practices</b> ולמה כדאי לישם אותם?
307	שכל מפתח מקצועי צריך להכיר <b>Best Practices</b>
307	בחירה שמות
309	<b>KISS</b>
310	<b>DRY</b>
310	לא להמציא את הגלגל
311	<b>Make it work, make it right, make it fast</b>
312	תיעוד
312	ניהול גרסאות
313	לבקש עזרה
313	ביקורת עמיתים – <b>Code Review</b>
314	<b>Tech Design</b>
315	<b>Best Practices</b> ואוטומציה
316	מה זה <code>?linting</code>
317	<b>ESLint</b>
318	רישמה של <b>Best Practices</b> והחוק הרלוונטי של <b>ESLint</b>
318	בל' מספרי קסם

319.....	השוואה קפדיות: ===
319.....	קוד לא נגיש .....
320.....	חלוקת לחלקים קטנים .....
320.....	אוריך מינימלי ומקסימלי לשימוש משתנים .....
321.....	בלי eval .....
322.....	בלי משתנים שלא הוצהרו .....
<b>323 .....</b>	<b>נספח: בדיקות, יציבות ואייבות קוד .....</b>
323.....	קצת רקע .....
324.....	מבנה בדיקות .....
325.....	בדיקות ייחודית .....
326.....	בדיקות קצה לקצה (End-to-End) .....
326.....	בדיקות ממשקי משתמש (UI Tests) .....
327.....	ספריות ופרימורקים מומלצים .....
328.....	סיכום .....
<b>329 .....</b>	<b>נספח: Corvid by Wix</b>
<b>329.....</b>	<b>הקדמה .....</b>
<b>329.....</b>	<b>מה בוניים? .....</b>
<b>330.....</b>	<b>איך מתחילה? .....</b>
<b>332.....</b>	<b>הצגת נתונים מסד הנתונים באתר .....</b>
333.....	Dataset .....
334.....	חיבור רכיבים למידע מהטבלה .....
<b>337.....</b>	<b>הוספת משימה חדשה .....</b>
337.....	Events .....
339.....	\$w .....
340.....	wix-data .....
340.....	wixData.insert() – הוספת רשומה לטבלה .....
<b>343.....</b>	<b>שינוי סטטוס המשימה .....</b>
344.....	wixData.get() – שילוף רשומה מהטבלה .....
345.....	wixData.update() – עדכון רשומה בטבלה .....
<b>346.....</b>	<b>מספר המשימות שלא הושלמו .....</b>
346.....	wixDataQuery .....
349.....	\$w.onReady() .....
<b>350.....</b>	<b>סינון המשימות לפי סטטוס המשימה .....</b>
351.....	wixDataFilter .....
<b>354.....</b>	<b>ניקוי המשימות שהושלמו .....</b>
354.....	wix-window .....
355.....	wixWindow.openLightBox() .....
356.....	wixWindow.lightbox.close() .....
357.....	wixData.bulkRemove() – מחיקת רשומה מהטבלה .....
<b>359.....</b>	<b>נספח – יצירת מסד נתונים .....</b>
362.....	Sandbox   Live .....

362.....	Permissions
<b>363.....</b>	<b>סיכון</b>
<b>364 .....</b>	<b>נספח: ג'אווהסקרייפט מונחה עצמים..... מה זה תכונות מונחה עצמים ?</b>
364.....	
370.....	Prototype based

## על הספר

הספר "לימוד ג'אוوهסקרייפט בעברית" מלמד את השפה ג'אוوهסקרייפט (JavaScript), שפת סקריפט קלה ופשטונה ללימוד שהפכה לפופולריות מאוד עם השנים. הספר מיועד ללא-מתכנתים שרצוים להתנסות בשפת תכנות ולמתכנתים בג'אוوهסקרייפט שרצוים להעמק את הידע התיאורטי שלהם.

הספר מתחילה בלמידה בסיסי של משתנים ומגיע עד לימוד תכונות אסינכראוני -AJAX. בכל פרק בספר יש הסברים תיאורתיים לצד דוגמאות רבות הממחישות את העקרונות השונים שהוסבו בו, ובסיומו תמצאו שאלות דוגמה לתרגול עם הסברים מקיפים על הפתרונות. בנוסף על הספר קיים אתר המוביל מאות תרגילים ופתרונות אשר יסייעו לכל הלומדים להציג שליטה בעקרונות השפה ובהחברה שלה. הספר מיועד להביאו אדם שלא מכיר את ג'אוوهסקרייפט לנקודה שבה הוא מכיר היטב את השפה ואת הת לחבריה שלה וידעו איך להשתמש בה כדי לפתור בעיות בסיסיות.

הספר יסייע גם למתכנתים מנוסים יותר שمبرקשים לחזק את הידע התיאורטי שלהם. יש בו הסברים על יכולות מתקדמות מאוד של השפה שהוחלו בשנת 2017, כמו מימוש טוב יותר של תכונות אסינכראוני, AJAX באמצעות `fetch` ותכונות נוספות.

## על המחבר

רן בר-זיק הוא מפתח תוכנה משנת 1996 ב{}{
 \begin{array}{l} \text{מג'וון שפות} \\ \text{פלטפורמות} \\ \text{ועובד בפתח בכיר במרכזי} \end{array}}{}  
פיתוח של חברות רב-לאומיות, מ-HP ועד Verizon, שם הוא מפתח בטכניקות מתקדמות הן בצד  
הלקוח, הן בצד השירות, ושם דגש על בניית תשתיות פיתוח נכונה, על שימוש ב-CD\CI וכמוון על  
אבטחת מידע.

נוסף על עבודתו בפתח במשרה מלאה, רן הוא עיתונאי ב"הארץ" במדור המחשבים, שם הוא מסקר  
מושאים הקשורים לטכנולוגיה ולאבטחת מידע וכותב על אינטרנט ורשתות.

משנת 2008 מפעיל רן את האתר "אינטרנט ישראל" (internet-israel.com), שהוא אתר טכני  
המכיל מדריכים, מאמרים וסבירים על תכנות בעברית ומתעדכן לפחות פעם בשבוע.  
רן נשוי ליעל ואב לארבעה ילדים: עומר, כפיר, דניאל ומיכל. רץ למרחקים ארוכים וחווב טולקין  
מושבב.

# על העורכים הטכניים

## דניאל שטרנלייבט

דניאל שטרנלייבט הוא מפתח Frontend מאז גיל 14, המיסד של המיזמים *Common Ninja* ו- *There is a bot for that*, צרכן בבד של קוד Open Source (ומשתדל גם לתרום בהזירה) ונכון לזמן כתיבת הספר מוביל את גילדת *Frontend* בחברת אוטברין. בנוסף על כך, דניאל הקים ומוביל את קבוצת הווטסאפ *Community FEDs* שמאגדת מפתחי Frontend מחברות מובילות בארץ ובעולם,ובה מעליים דיוונים, מתיעצים ומשתפים לינקים, חדשות ועדכונים מעולם ה-*Frontend*. בעבר כתוב בבלוג "עיצוב גרפי וטכנולוגיה", וכיום הוא כותב בלוג טכני על טכנולוגיות *Frontend* ועל *NodeJS*. בשאר הזמן הוא נשוי לרונה ואבא להדר ולאיל המתוקים, מתופף, גולש סקי, משחק כדורים, צופה בסדרות, קורא ספרי פנטזיה ומתח ומדקלם את כל סרטיו דיסני בעלפה.

## gil Fink

gil Fink הוא מומחה לפיתוח מערכות ווב, Google Developer Expert, Microsoft Developer Technologies MVP. ביום הוא מייעץ לחברות ולארגונים שונים, שם הוא מסייע בפיתוח פתרונות מבוססי אינטראנט ו-SPAs. הוא עורך הרצאות וסדנאות ליחידים ולחברות המעוניינים להתמחות בתשתיות, בארכיטקטורה ובפיתוח מערכות ווב. הוא גם מחבר של כמה קורסים רשמיים של מיקרוסופט (Microsoft Official "Pro Single Page Application Development" Course - MOC), מחבר משותף של הספר "Full Stack Frontend" (Apress) ושותף בארגון הכנס הבינלאומי DUPAngular. פרטים נוספים על גיל: <http://www.gilfink.net>

## יגאל סטקלוב

יגאל סטקלוב הוא מפתח Frontend-i-*Full Stack*, מוביל טכנולוגי ומנהל פיתוח מנוסה בעל ותק של יותר מעשור וחצי בתעשייה. ביום משמש מנכ"ל חברת *Webiya*. בעבר היה מובילי תחום *Frontend* בחברות *Wix* ו-*Netcraft* והוביל פרויקטי פיתוח רבים. יגאל פועל מאוד למען קהילת *Frontend* בארץ והוא אחד מהיזמים והמארגנים של כנס ה-*Frontend* הבינלאומי הראשון בישראל, בסיס *YGLF* (You Gotta Love Frontend).

## תום ביגלאיזן

מפתח Frontend ומיצב, נשוי, אב לשתיים וגר בתל אביב. תום התחיל את דרכו בתחום ה-Frontend בתחילת שנות האלפיים מכיוון פחות צפוי – בפיתוח אפליקציות ביג'אווהסקריפט לממירים של ייס בסטארט-אפ קטן. אחרי שזה נסגר, עבד בעצמאות ובכמלה סטארט-אפים, הפרק לאחד המומחים הראשונים בארץ ל-HTML ו-CSS ולא היה בקיא ממנה באגים של CSS עם `rtl` באקספלורר 6. תרם לפרויקטים שונים בקוד פתוח, היה שותף בגין פלטפורמת ניהול התוכן דרופל ובנה את אחד הטמפלטים העבריים הפופולריים באותה תקופה. תום גם עיצב את האיקונים שעדיין נמצאים בשימוש עד היום בנגן הוויידיאו הפופולרי בקוד פתוח VLC. לפני קצר יותר משבע שנים נחת ב-Wix, ובהמשך השנים האחידנות עמד בראש צוות קטן ומוכשר שמוביל את תחום ה-Frontend במדיה – וידיאו, תמונות, וקטורים ואנימציות בפלטפורמת בניית האתרים של Wix.

## צחי נמני

צחי נמני הוא מומחה בפיתוח, סרבר, אוטומציה ודב אופס מתמחה בטכנולוגיות הבאות: Java, Kotlin, Node.js, C#, Docker, Kubernetes, Git, Terraform, Selenium, Appium, Cypress. ביום הוא עבד בחברת הייטק בתחום התיירות. בנוסף, צחי מיעץ ומרצה לחברות וЛИחידים בנושאים של Docker, Kubernetes, Automation Development, Docker, Kubernetes CI/CD, Docker and Kubernetes. ברגע הוא עבד על סדרה של הרצאות בנושא צחי מאמין גדול בקוד פתוח ותרום לפרויקטים שהוא משתמש בהם.

## על ג'אווהסקריפט

ג'אווהסקריפט החלה את צעדיה הראשוניים ב-1993 בשפה שפועלת בסביבת דפדפן וונועה להעшир דפי HTML. בג'אווהסקריפט יכולנו ליצור אнимציות ופידבק למשתמשים – כל דבר שהיה קשור לאתר או אינטרנט דינמיים, למשל דברים שנראים היום מאוד טריוויאליים ופשטועים כמו לחיצה על כפתור שעושה פעולה בלבד בדף. אף על פי שההתחלת שלה הייתה צנואה, ברנדון אייר, ממציא השפה, יצר אותה מלבת חילה בשפה גמישה מאוד. הגמישות זו, וגם חוסר ההבנה של רבים מהמתכנתים שהשתמשו בה בוגרנו לעקרונות הבסיסיים שלה, גרמו ללא מעט מתכנתים בשפות אחרות לחדול בה. גם השם שלה לא סייע לתדמית. השם ג'אווהסקריפט נקבע מסיבות שיווקיות בלבד – JAVA היא שפת תכנות פופולרית, ואנשי נטסקייפ חשבו שכיר יוסיפו לתדמיתה. בפועל השם זה לא ממש עדיר, ואין כמובן שום קשר בין ג'אווה לג'אווהסקריפט.

על אף ההתחלה הקשה, ג'אווהסקריפט הפכה לפופולרית מאוד. בשנת 1996, חברת נטסקייפ העבירה את השילטה בסטנדרטים של השפה אל ארגון ECMA, ארגון אירופי (היום בינלאומי), המתמחה בתקינה. המהלך הוביל לשחרור הסטנדרט של השפה, שידוע בשם ECMAScript, וג'אווהסקריפט "התיירה" לפי התקינה של ECMAScript. בשנת 1997, שנת שחרור ECMAScript, ג'אווהסקריפט, כפי שהיא מישמת בדפדכנים שונים, עוקבת אחר התקינה של ECMAScript, שהיא בעצם "תוכנית המתאר", וג'אווהסקריפט עצמה היא היישום. לבסוף יש מספר משלה בצד מילימ'ס ES (ראשי תיבות של ECMAScript).

מיקרוסופט התנגדה בתחילת ליישום השפה ויישמה שפה משלה בשם Jscript בשם Jscript בדפדפן אינטרנט אקספלורר, שהיא בדומה לג'אווהסקריפט. למרות היריבות הגדולה בין אנשי מיקרוסופט לאנשי ECMA, שנוצרה בתוצאה מפיתוח שתי שפות שונות על שני תקנים מתחברים, העקרונות של ג'אווהסקריפט שולבו גם בגרסתו של מיקרוסופט. הפופולריות של השפה עלתה באשר מקרומדיה (יווצרת פלאש) שיתפה פעולה עם ארגון ECMA ושילבה את עקרונות השפה בשפת Actionscript, ששימשה את תוכנת פלאש שהייתה פופולרית מאוד אז.

בשנת 2008 נפגשו אנשי מיקרוסופט ו-ECMA באוסלו, שיחות השלום האלו הסתימו בהצלחה. תקן ES5, הגרסה הרביעית של ג'אווהסקריפט, שוחרר ויושם בכל הדפדכנים שהיו קיימים אז.

מאז, התפתחות השפה והתפוצה שלה הואצז דרמטית. דפדפן כרום, שמריץ ג'אווהסקריפט באופן יוצא דופן, נכנס אל השוק בסערה ואפשר למפתחי ג'אווהסקריפט לבתוב סקריפטים שפועלים על מנוウ 87 העצמתי של כרום ולהריץ ג'אווהסקריפט במהירות מסחררת. השימוש ב-AJAX תקשורת אסינכורונית עם השרת – נכנס לפעולה, החליף שיטות מישנות בגין Long polling ואפשר לתארים לספק חוות שימושיות מדיימות למשתמשים. בשנים האחרונות, פרימורקים וספריות ג'אווהסקריפט אפשרו פונקציונליות מורכבת מאוד וספריות אחרות אפשרו כתיבה של ג'אווהסקריפט גם לטלפונים ניידים ואףלו בטלפון. הראשונות שבספריות האלו נקראו MooTools ו-jQuery והן אפשרו לכל מתכנת לבתוב אפליקציות בקלות. הספריות האחרונות נקראות ריאקט, אנגולר ו-ענ

והן מאפשרות לבנות תוכנות מורכבות מאוד על גבי הדפדפן (צד הלוקה). ג'אווהסקריפט לא נותרה מוגבלת רק לצד הלוקה, בולומר לדפדפנים ולמబשרי קצה אחרים; המימוש של ג'אווהסקריפט לצד השירות, הידוע בכינוי `node.js`, הפרק לפופולרי גם בשרתים. ג'אווהסקריפט מರיצה ביום אפליקציות מורכבות גם לצד השירות, במיוחד אפליקציות שצרכות לבצע קראיות ולשרות מיליוןית משתמשים. ביום אפשר למצוא ג'אווהסקריפט בכל מקום: באתר אינטרנט, באפליקציות של טלפונים ניידים, באפליקציות המיעודות למחשבים רגילים וכמוון בשרתים. הביקוש למתכנת ג'אווהסקריפט נמצא בשיאו ואין זה פלא – אפשר לעשות בשפה זו המונע דברים יישומיים כמעט ממאפס. יש כל הרבה ספריות וכל עזר, עד שבעת שבוע יוצאה ספריה שימושית חדשה. בעזרה ידע מועט אפשר לעשות הרבה מאוד. מה שחשוב הוא ידע בסיסי בשפה.

בשנים האחרונות, תקן ES מתעדכן בכל שנה ומתווספים אליו תכונות ושימושים חדשים. ספר זה מעודכן לגרסה الأخيرة של ECMAScript. חשוב לציין שאם התקן מתעדכן, אין פירוש הדבר שהעדכון החדש מופיע מיד בדפדפנים שמריצים ג'אווהסקריפט או בשרתים שמריצים ג'אווהסקריפט, אלא לוקח זמן עד שהעדכנים החדשים ביותר עושים את דרכם אל הדפדפנים/שרותים שוכלו ממשתמשים בהם. אם שמעתם מפתחי אינטרנט "מקטרים" על דפדפנים **ישנים** – זו בדיק הסיבה.

זה המנייע לבתיבת הספר. הבן שלי, ביום מתכנת בזכותו עצמו, ניסה ללימוד ג'אווהסקריפט מאפס ולא הצליח למצאו ספרים בעברית. החומר שיש ביום בעברית בנוגע לג'אווהסקריפט הוא דל ומיושן. חלק מהחברות העזר הנמצאות בבתי הספר מתייחסות לתקנים שהפסיקו להיות בשימוש בשנת 2007! התיחסות אמיתית לתקנים החדשניים ביותר של השפה, שיצאו בשנת 2017, אין בנמצא בעברית. התחלתי כתוב הספרים עבור בני, ומפה לשם הבנתן שאני חייב לקחת את זה הלאה.

ג'אווהסקריפט היא שפה שקל ללימוד. בניגוד לשפות אחרות, לא נדרש בה סביבת שרת מורכבת או כל פיתוח שעולים בסף. לא נדרש ידע מוקף במידע המחשב. כל שצריך הוא לפתוחepad במחשב, לפתח דףדף ולהתחליל ללמידה ולפתחה. צרי גם הדרכה נcona ומשמעות עצמית. אני מקופה בספר זהה תמצאו לפחות הדרכה נcona. המשמעות העצמית – עלייכם. אני מאמין שכך שתהתקדם בספר תראו את פירות הלימודים, הניצוץ בעיניכם יתחזק ולא תצטרכו עוד ממשמעת עצמית – אתם פשוט תתאהבו בג'אווהסקריפט בכלל ובפיתוחו לוב בפרט. אל תלגו על הפרק שבו אני מסביר איך למדוד; זהו הפרק החשוב ביותר בספר.

אני מאמין לכם הצלחה רבה, בין שאותם מתכנתים בתחילת דרככם לבין שאותם מתכנתים ותיקים שימושיים בספר כדי לשפר את הידע שלכם.

– רן בר-זיק

# AIR לומדים

לא למדתי מדעי המחשב באוניברסיטה או במכללה. למען האמת, עד גיל מאוחר מאוד לא למדתי תכנות בעזרת מדריך. רוב מה שאני יודע למדתי לא הדרכה, וככמה סיבות: הראשונה היא שכאשרעשיתי את צעדי הראשוניים בתכנות, בשנת 1996, החומר שהיוה זמין באינטרנט היה דל מאוד. על חומר בעברית לא היה מה לדבר, והחומר באנגלית סיפק בדרכו כלל רק את הדוקומנטציה. אני לא ממש מתגאה בכך; למידה בלבד לא הדרכה היא למידה מאוד לא יעילה. הדרכה ממשמעה לא רק מרצה מנוסה, אלא גם אתר אינטרנט שבו יש הסבר מקיים, פורום או קבוצה בראשת חברותית זו או אחרת (לא רק פיסבוק), שיש בהם אנשים מנוסים שיכולים לסייע או להפנות לחומר עזר. היא גם מקום שבו אפשר לתרגל ולהתנסות. לרובו המזל, בימים אלו קיימים שלל חומרים, עזרה וסייע. גם הספר הזה הוא הדרכה. לא תידרשו לצול לתוכה הדוקומנטציה של ECMAScript על מנת להבין את השפה, אבל גם בעזרת הספר תמצאו דרך טובה ללמידה. ביוון שרוב הזמן למדתי לא הדרכה, גיבשתי במה עקרונות למידה שהבנשתי לספר זהה. אני ממליץ לכם לעקוב אחריהם.

## ORGANO לעצמכם סביבת עבודה מסודרת

הפרק הראשון עוסק במבנה סביבת העבודה והוא הפרק החשוב בספר. ארגנו את המחשב שלכם וסדרו לעצמכם תיקיה מאורגנת שבה תשמרו את כל התרגילים. אם המחשב שלכם מופיע התראות של עדכוני ג'אווה או משווא בסגנון דומה, טפלו בכך. וDAO שאתם יכולים להיכנס לאתר הלימוד שמלואה את הספר ושהסήמה שלכם תקינה ופועלת.

## קראו את הפרקים לפי הסדר

הפרקים לא פוזרו באקראי אלא תוכננו בסדר מסוים. אין טעם ללמד AJAX לפני שambilנים AIR תכנות אסינכרוני עובד. אין טעם ללמד תכנות אסינכרוני לפני שambilנים AIR קולבקים עובדים. ואם גם זה נשמע לכם ג'יבריש, סימן שאתם צריכים להתחיל מההתחלתה. קראו כל פרק לפי הסדר.

## קראו כל פרק פעמיים וכתבו לעצמכם את כל הדוגמאות

קראו את הפרק פעם אחת קרייה שוטפת. לאחר מכן קראו אותו שוב. הפעם קחו את כל דוגמאות הקוד, העתיקו אותן לסביבת העבודה שלכם וחקקו בהן! נסו לשנות את הערכיהם, לגרום לשגיאות, לכתבם קוד דומה.

אנו מאמין שקוד לומדים דרך הידים ולא רק דרך הראש. חשוב להבין את התיאוריה ואת הרעיונות מאחורי מה שמנסים לעשות, אך ללא מימוש, הידע הזה יתנוון ויעלם, בדיק במו בשפת דיבור. אם לא תשתמשו ותתרגלו, גם הלימוד התיאורטי המעמיק ביותר לא יהיה שווה הרבה. הקרייה הראשונה

נوعדה ללימוד התיאוריה, הקריאה השנייה نوعדה לתרגול. לימוד שפה הוא לא מרוץ! קחו את הזמן, כתבו את כל דוגמאות הקוד ונסו לכתוב באלו משלכם.

## פתרו את התרגילים לדוגמה בסוף כל פרק

בסוף כל פרק יש תרגילים לדוגמה. נסו לפתור אותם. אל תתייאשו מהר ואל תורצו אל הפתרון אלא שברואו קצת את הראש. הצלחתם לפתור? נזהר. קראו את הפתרון המוצע ואת ההסביר וראו אם הם דומים לשכלכם או שונים במקצת. יש יותר מפתרון אפשרי אחד לכל בעיה...

## התיעצו עם אחרים

יש לא מעט קבוצות בעברית (בפייסבוק, אבל לא רק) המיעודות ללימוד ג'אווהסקרייפט ולדיאו בה. מצאו את זו שהכى נוח לכם בה. אל תהססו לשאול שם שאלות. לא הבנותם משהו? דוגמה בלשחי לא הייתה מובנת? הפתרון לתרגיל לא היה ברור מספיק? שאלו שם, ובערבית. אל תהססו לקרוא, למשל דיאו על משמעות השפה, להשתתף בדיונים או לסייע למי שהיודע שלו דל משלכם. אל תשחחו להבין את רוח הדברים בקבוצה ולא להציג או להעיק. רוב המשתתפים בקבוצה הם אנשים עובדים שלאו דואקה זמינים להודעות מיידיות.

## הגיעו לسدנאות ולמייטאפים

לא מעט חברות מארגנות בחינם סדנאות, מייטאפים ומפגשים שבהם מתכנתים מציגים את ג'אווהסקרייפט ומרצים עליה. כדאי מאוד להגיע למפגשים האלו, לא רק על מנת לשמע את התכנים ולהשתתף בסדנאות אלא גם להכיר אנשים אחרים בתעשייה. עם חלוקם אתם תתכתבו בקבוצות הפיסבוק לג'אווהסקרייפט. קהילת מפתחי הג'אווהסקרייפט בארץ היא קהילה מאוד שיתופית וקרובה, ורבים בה מכירים זה את זה.

## תרגול, תרגלו, תרגלו

הספר לא שווה הרבה הרכבה בלי תרגול מקיים ושימוש בו. אל תעברו לפרק הבא לפני שתסינו את כל התרגולים שקשרוים לפרק שקראתם. זה לא קרייטי, אלא סופר-קרייטי.

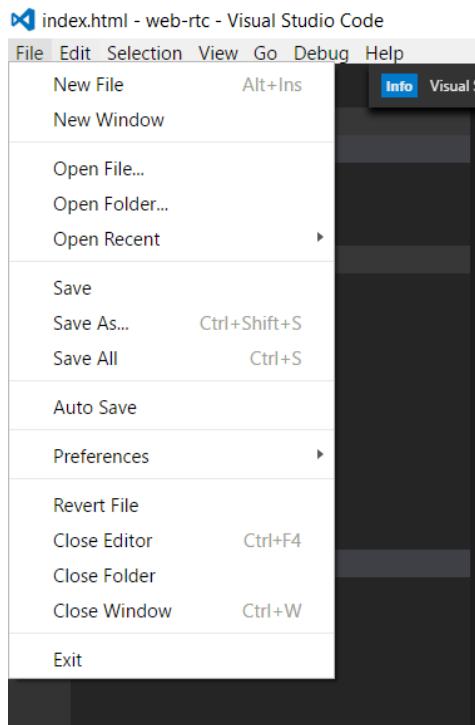
## התקנת סביבת העבודה ודרך הלימוד

ג'אווהסקריפט יכולה לrhoץ בסביבת שרת או דרך הדפדפן. איך זה בדיקןעובד? ג'אווהסקריפט נמצאת בקובץ טקסט. כן, בדיקן כמו זה שאפשר ליצור באמצעות בתbn הטקסט (Notepad) שיש בכל מערכת חלונות. בדפדפן מותקן כל שילוקה את קובץ הטקסט זהה וMRIץ אותו ואת הפוקודות שנמצאות בתוכו. אם הדפדפן היה אדם, הוא היה פותח את קובץ הטקסט וקורא את מה שיש בתוכו, למשל: "ליר ימינה ופתח את הדלת", ועשה בדיקן מה שכתבו. הפעולה הזאת נקראת בלשון הפופולרית "רינדור", מלשון *render* בלעדי. הדפדפן לוקח את קובץ הג'אווהסקריפט וMRIץ אותו. קובץ הג'אווהסקריפט יכול לעשות כל מיני דברים ולהציג או לא להציג אותם.

איך הדפדפן טוען את קובץ הג'אווהסקריפט? יש כמה דרכים לעשות זאת, אבל ברגע ארצתה למדם אתם איך לכתב קובץ ג'אווהסקריפט וראות אותו פועל על מנת להבין את כל השפה וכתבו משהו באופן ראשוןו ביותר. החלק החשוב והקשה ביותר הוא יצרת סביבת העבודה, בלומר סביבה מומוחשבת שבה אפשר להקליד ג'אווהסקריפט וראות אותה עובדת. סביבה זו היא חשובה מאוד כאשר לומדים, כיון שלימוד של שפת תכנות נעשה ראשית בול "דרך הידיים" וחוובן מאוד לא רק לקרוא אלא גם לתרגל. וכך לתרגל צריכה סביבה שמאפשרת להקליד פוקודות שפה, לשמרן וראות את הפלט. אני שב ומדגיש: התקנת הסביבה היא החלק הקשה ביותר ביסודו תחילת לימוד שפה חדשה וגם החשוב ביותר. לפיכך כדאי בסבלנות, לקחת נשימה ארוכה ולזכור שדוקא עבשו מתמודדים עם החלק הקשה ביותר.

הבה נתחל בעורך טקסט טוב. אמנם אפשר להשתמש בנוטפ, אבל הוא לא מציע צביעת קוד לצורך עזרה בקוריאות ולא יצרת הדוחות בקלות. יש כמה עורכי טקסט המותאמים במיוחד לבתית ג'אווהסקריפט, שאציגו כמה מהם. בחרו בעורך טקסט אחד!Robom Zeimim למדים ומכלים יכולת עריכה בסיסית של HTML, CSS וג'אווהסקריפט.

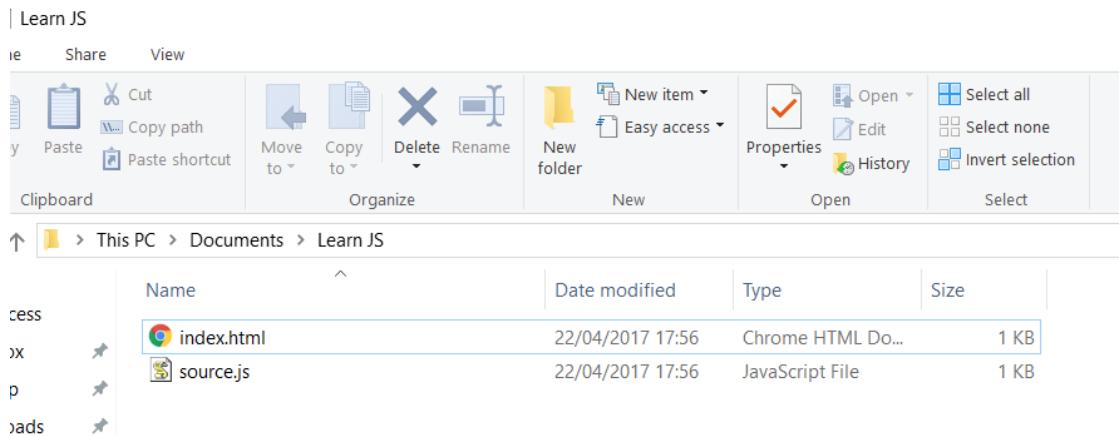
**שימוש לב:** מתקנים מנוסים לא משתמשים בעורך הטקסט הפשוט אלא בעורך טקסט משוכל יותר שנקרא IDE או Integrated Development Environment – סביבת פיתוח מושלבת. הסביבה הזאת מאפשרת השלמת קוד, מצינית שגיאות בכתיבה ו�� יכולה להריץ את הקוד עצמה. IDE מעולה הוא Visual Studio Code. הוא חינמי, מובוסס קוד פתוח, כתוב בג'אווהסקריפט (כן, כן), נתמך על ידי מיקרוסופט וניתן להורדה פה: <https://code.visualstudio.com/> אחרי התקינה תוכלו לפתוח את התוכנה, לבחור ב-File ואז לפתח את התיקייה שבחרתם ולפתח או ליצור בה קבצים. אני ממליץ לכם בחום להוריד את קוד זהה: הוא חינמי לחלוטין ואני מככיך על המחשב.



IDE מוצלח אחר הוא Atom. גם הוא חינמי וمبוסס קוד פתוח וגם הוא... בתוכו בג'אוوهסקריפט. הוא נתמך על ידי גיטהאב ונitin להורדה מה: <https://atom.io/>. הוא דומה ל-Visual Studio Code ואחרי ההורדה והתקינה שלו אפשר להפעיל אותו בקלות. הוא עורך טקסט נוסף שנחשב לאמין וטוב הוא תוכנה חינמית בקוד פתוח שנקראת ++NotePad. הוא ניתן להורדה בקישור הבא: <https://notepad-plus-plus.org/download> והוא בסיסי יותר מאשר קודמי.

שימוש ב-Visual Studio Code או ב-Atom הוא מומלץ יותר כיון שיש בו "השלמה אוטומטית" של פקודות נפוצות בג'אוوهסקריפט, דבר המקל מאד את הלמידה. כמו כן הוא מציג שגיאות בקוד בברכוב הכתיבה, עד לפני הרצה. הסביבה הטובה ביותר ללימוד היא יצירת קובץ HTML שטוען קובץ ג'אוوهסקריפט. קובץ HTML הוא קובץ שהדף יודע לפרש ולהציג, והוא יכול לקרוא לקובץ ג'אוوهסקריפט באופן דומה שהדף יrnder אותו. כאמור, rnder הוא הרצת הפקודות שנכתבות בקוד ג'אוوهסקריפט והציגן על המסך או במקום אחר.rnder, מילשון render, הוא מונח מתחום מדעי המחשב ופירושו הוא "הרצה". בשאני כתוב "הקוד מrnder" הפירוש הוא שהקוד רץ ומציג את התוצאות. יש ליצור במחשב תקיה – זה יכול להיות ב"המסמכים שלי" או על שולחן העבודה – ובתוכה ליצור קובץ בשם source.js וקובץ בשם index.html.

בחלונות יצירת תקיה נעשית באמצעות: כניסה אל סיר הקבצים, בחירת המქם שבו רוצים למקם את התקיה. לחיצה על המקש הימני של העכבר ואז בחירה ב"חדש" וב"תיקיה". את הקובץ עצמו ניתן ליצור באמצעות התוכנות Visual Studio Code או Atom או אפילו ++NotePad. פיתחו את אחת התוכנות האלה (אני ממליץ על Visual Studio Code) נווטו אל התקיה ובחרו ב-File ואז New.



ודאו שמערכת החלונות או המק שלכם תומכת בתצוגת שם הקובץ המלא, כולל הסיומת (.index.html.txt) של הקובץ. אחרת, קובץ ה-.html יהיה בעצם (.index.html)

בקובץ ה-HTML כתבו את הקוד הבא:

```
<!doctype html>
<html>

<head>
  <meta charset="utf-8">
</head>

<body>
  <script src=".source.js"></script>
</body>

</html>
```

בתוך קובץ ה-.js.source כתבו את הטקסט הבא:

```
document.write('Hello World!');
```

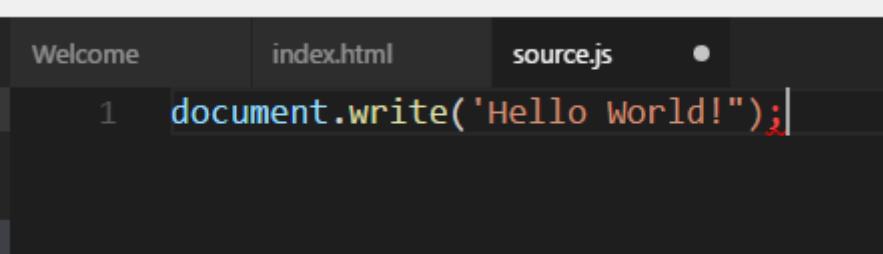
אחרי לשמורת את תוכן שני הקבצים, פתחו את הקובץ בדף כרום או בפיירפוקס (לא באdeg'). אם הכל תקין, תראו שכותוב על המסך "Hello World" – כתבתם את הג'אויסקייפ הראשון שלכם!

שיםו לב: זה השלב המועדף ביותר לפורענות, שועל להיות מתסכל מאוד, אבל הוא שלב חשוב ביותר לאסור להרים ידים ולהתיאש. אם פתחתם את הקובץ ודבר לא הופיע, נסו את הדברים הבאים:

בדקו שacky קראתם לקבצים index.html ו-.js.source. הבדיקה צריכה להתבצע באמצעות הקפתו הימני של העכבר בחלונות, כדי למנוע מצב שבו מערכת הפעלה הוסיפה תוספות לשם ו-index.html נשמר בשם index.html.txt.

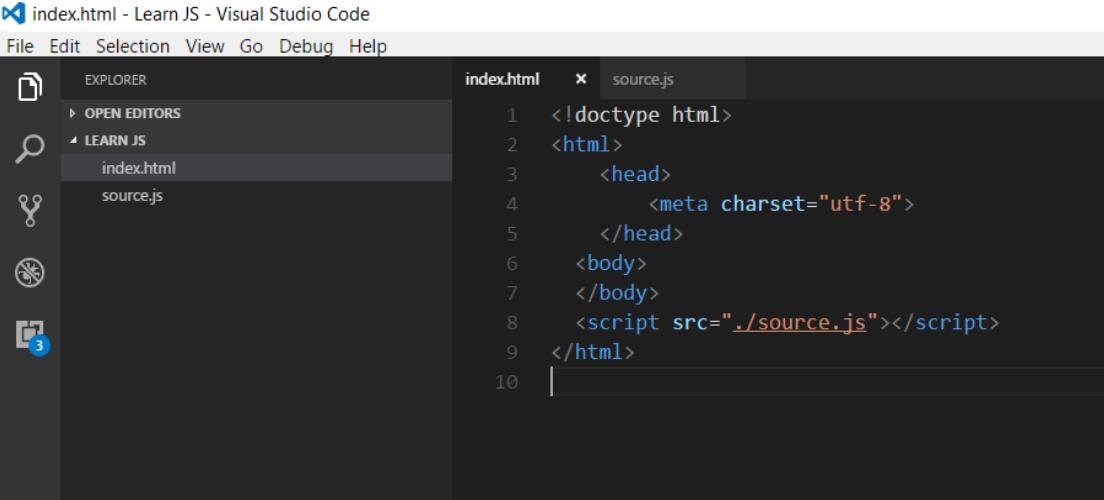
בדקו שגם פותחים את הקבצים בדף כרום או בפיירפוקס. בדקו שאין תוספים מיוחדים לדפינים שעולמים לחסום את הרצת הקובץ על ידי הריצה של מצב פרטיות.

בדקו שהקלדתם את הטקסט בשורה בקובץ `source.js` ללא רווחים או תווים מיוחדים. נסו להשתמש ב-Atom או ב-Visual Studio Code. שימוש לב שאין התראות על שגיאות הקלדה. באנו לمثال מובאת דוגמה של שגיאת הקלדה שביצעת. אם קיבלתם התראה כזו, בדקו שוב שלא טעתם בגרש ושלא הכנסתם רווחים מיוחדים כמו בדוגמה זו שבה יש שימוש שגוי בגרש יחיד ואז בגרשיים. עדיף להשתמש תמיד בגרש יחיד:



```
document.write('Hello World!')
```

אם אתם משתמשים ב-Visual Studio או ב-Atom, סביבת כתיבת הקוד שלכם אמורה להיראות כך:



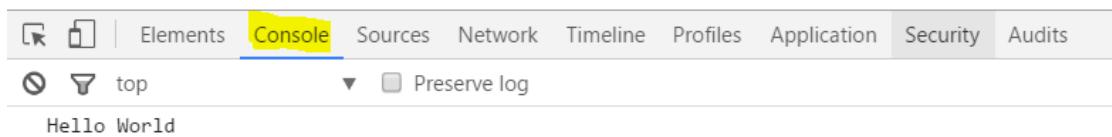
```
index.html
1 <!doctype html>
2 <html>
3   <head>
4     <meta charset="utf-8">
5   </head>
6   <body>
7   </body>
8   <script src="./source.js"></script>
9 </html>
10
```

מצד שמאל תוכל לראות את כל הקבצים בתיקייה. כדאי ליצור תיקייה מיוחדת בשם `learnJavaScript` או בשם דומה ולא לשים את כל הקבצים בתיקייה משותפת כמו "המסמכים שלי".

מצד ימין תוכל לראות תוכן הקבצים. בצד שמאל רואים את תוכן הקובץ `index.html`. אם תקלידו דבר מה, תוכל לראות שיש השלמה אוטומטית של קוד HTML, ואם תקלידו בקובץ `source.js` שמכיל את היג'אווהסקריפט תראו שיש השלמה אוטומטית של פקודות ג'אווהסקריפט. נוסף על כך, תקבלו גם התראה על קוד לא תקין.

כיוון שאתה כבר מפתחי ג'אווהסקריפט מקצועיים, כדאי שתתלווה לשימוש בקונסולה של הדפדפן. מדובר במקרה מיוחד מושך שמאפשר "לדבג" את ג'אווהסקריפט. הפעול "לדבג" כוננו להסתכל על צפונות הרינדור וראות ממש את הפלט של השפה או את תוכאות הרצאה שלה. בלומר מה שבתנו. נשמע מסובך? יש להבין איך הדפדפן מנסה להריץ את הקוד שלו (באמור מה שנקרא "רינדור", מילשון הרצאה, באנגלית) ולקבל מידע מידי במקרה שהוא נכשל, בלומר "זורק" שגיאה צבועה באדום בקונסולה ואפ' מפנה לשורה הביעיתית.

על מנת לראות את הקונסולה, פתחו את כל המפתחים של הדפדפן. בכרום ובפיירפוקס לחצו + i + Ctrl Shift אם יש לכם חלונות או + Cmd + Shift אם יש לכם מק. אפשר לעשות את זה גם דרך התפריט העליון בשני הדפדפנים. לאחר פתיחת כל המפתחים, לוחצים על לשונית Console.

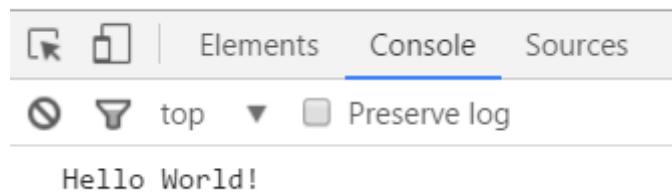


הבה נבדוק מה אפשר לעשות בעזרת הקונסולה. היבנוו אל source.js וחליפו את הטקסט ל:

```
console.log('Hello World!');
```

טענו מחדש את הדף באמצעות Ctrl + F5 או Cmd + R במק. הטעינה מחדש מחדש חשובה. הדפדפן לא יודע שהוכנסו שינויים בקובץ הג'אוوهסקריפט, ויש לגרום לו לטען מחדש את קובץ הג'אוوهסקריפט על מנת להריץ את הפקודות חדשות. אחרי הטעינה מחדש הסתכלו על לשונית Console.

אם הכל תקין, המסר יהיה ריק, אך בקונסולה תראו Hello World! יש!



**חשוב:** אל תלגו על השלב הזה. בכל שלבי הלימוד כדאי להשתמש בקונסולה, שהיא הרבה יותר נוחה להצגה. אם משאנו לא עובד, אנו בדקו את הדברים הבאים:

1. האם שלמתם את שלב הקוד? הצלחתם להציג Hello World על המסך?
2. האם שמרתם את הקובץ לאחר השינויים?
3. האם טענתם מחדש באמצעות Ctrl + F5 או Cmd + R את הדפדפן?

מדובר בסביבת העבודה ביתובה ביותר ללימוד ג'אוوهסקריפט, אך יש סיבות לעבודה נוספת. בראשת יש אתרים המאפשרים לבתוב ג'אוوهסקריפט ישירות, להריץ את הקוד דרכם ולראות את התוצאות. אתר מפורסם וחשוב זה הוא <https://codepen.io/> ואפשר להקליד בו פקודות של ג'אוوهסקריפט. פתחו שם חשבון וצרו "pen" חדש. בדקו שיש אפשרות להכניס קוד HTML, CSS ו-JS, שהוא בעצם קיצור של ג'אוوهסקריפט. אפשר להכניס את הקוד ולראות את התוצאות על גבי דף מדומה או על גבי הקונסולה של הדפדפן.

מכאן, דרך הלימוד תהיה פשוטה למדי. אני אסביר על תכונות מסוימות של השפה ונתן דוגמאות. מומלץ בחום רב להעתיק את הדוגמאות אל קובץ source.js ולהריץ את הג'אוوهסקריפט כדי לראות איך זה עובד באמת.

בסוף כל פרק יש תרגילים לתרגול עצמאי ומומלץ מאוד לנסות אותם בסביבת העבודה. אי-אפשר ללמידה שפה על ידי קריאה תיאורטיבת בלבד ורצוי לתרגם, לתרגם, לתרגם. את התרגול עושים רק בסביבת עבודה יציבה. לפיכך, אנא אל תדלגו אל הפרק הבא לפני שיש לכם סביבת עבודה יציבה. מומלץ מאוד לעבוד ב-code Visual studio code החינמית.

**תרגילים:**

במקרה "!Hello World" גרמו לקונסולה להדפיס את המילים "Ahla Bahla".

**פתרונות:**

היכנסו לקובץ `source.js`, מחקקו את הטקסט שיש שם והדביקו במקומו את:

```
console.log('Ahla Bahla');
```

שמרו את הקובץ, פתחו את הקובץ `index.html`, שטוען את הקובץ `source.js`, ורעננו את הדף. לחצו על `Ctrl + Shift + C`, לחצו על הלשונית `Console` וראו את התוצאות.

**תרגילים:**

גרמו לקובץ ה-HTML לטען קובץ ג'אווהסקריפט בשם targil.js ושמדפיס בקונסולה: new file.

**פתרונות:**

צרו קובץ targil.js באותה תיקייה של הקובץ index.html. יש לוודא שהוא בשם האמתי של הקובץ ושמרכיבת הפעלה לא מצמידה לקובץ עם הסיומת .txt. את זה עושים על ידי בדיקה בהגדשות התצוגה של מערכת הפעלה. פתחו את קובץ ה-HTML בעזרת עורך טקסט (מומלץ להשתמש ב- Visual Studio Code) ושנו את שם הקובץ הג'אווהסקריפט ל-targil.js.

```
<!doctype html>
<html>

<head>
  <meta charset="utf-8">
</head>

<body>
  <script src="./targil.js"></script>
</body>

</html>
```

בקובץ targil.js כתבו את השורה זו:

```
console.log('I am new file');
```

פתחו את הקובץ index.html בדפדפן ולחצו על מנת להציג את כל המפתחים. בחרו את הלשונית Console ובחנו את התוצאה.

פרק 1

# משתנים



## משתנים

החלק הבסיסי והחשוב ביותר הוא המשתנה. מדובר ברכיב שיכל להכיל בתוכו מידע, ואפשר לשנותו – זו הסיבה שהוא נקרא "משתנה". משתנה בג'אוوهסקרייפט מוגדר באופן הבא:

```
let variant;
variant = 'Hello World';
```

מה קורה פה? יש כאן הגדרת משתנה בשם `variant`. ההגדה נעשית באמצעות המילה השמורה `let`. מילה שומרה היא מילה מיוחדת בשפה שהשימוש בה שומר למשך מסויים. במקרה זה, `let` שומרה אך ורק להגדרת משתנים.

לאחר מכן מכנים ערך למשתנה. הפעולה זו נקראת "הצבת ערך" או "השמה", והוא נקראת כך מכיוון שהערך מוצב בתחום המשתנה. במקרה זה מדובר בטקסט הכלל את המילים `Hello World`. **שימוש לב:** יש סימן ";" בסוף כל שורה. בג'אווהסקרייפט מועב להציב סימן ";" בסוף כל שורה כדי למנוע בעיות וכדי שהסקריפט ישבוד. הוא מסמן סוף שורה עבור מנוע הג'אווהסקרייפט שמרנדר את הסקריפט, ממש כמו נקודה בסוף משפט. על אף שלא כל מנוע מקפיד על כך, אני ממליץ לכם:

הקפידו תמיד להקליד ";" בסוף כל שורה.

אפשר לקצר ולהציב את הערך מייד בהגדרת המשתנה:

```
let variant = 'Hello World';
```

הבה נבדוק את המשתנה ואת הערך שלו. אפשר להציב את המשתנה הזה בתחום `console.log` כפי שŁמדנו בפרק הקודם:

```
let variant = 'Hello World';
console.log(variant);
```

אם תציצו בקונסולה, תראו שמודפס המשפט "Hello World". מדוע? כיון שהוא מה שיש בתחום המשתנה. מן הסתם, משתנה ניתן לשינוי. נסו את הקוד זהה:

```
let variant = 'Hello World';
variant = 'I am a new version';
console.log(variant);
```

מה לפि דעתכם ייצג בקונסולה? ייצג "Hello World" או "I am a new version". למה? כיון שהערך הקודם "נדرس" על ידי הערך החדש. הכנסתם (בלומר הצבתם) ערך חדש לתוך המשתנה, ועכשו מה שיש בתוכו השתנה. כמשמעותם את המשתנה רואים את הערך החדש. יש הבדל מהותי בין הגדרת משתנה לבין הכנסת ערך לתוכו. הגדרת משתנה היא כמו בניית ארון או קופסה ואפשר בכל פעם להכניס לתוכו ערך אחר.

**שימוש לב:** אחרי שימושו מסויים הוגדר, או-אפשר להגדיר אותו מחדש. הקוד הבא:

```
let variant = 'Hello World';
let variant = 'I am a new version';
console.log(variant);
```

יגרום לשגיאה הבאה: **Identifier 'variant' has already been declared**. שמות המשתנים יכולים להיות מגוונים אך יש להם כמה כלליים מחייבים. אפשר להשתמש בכל אות שהיא ובסימנים \_ או \$ בתחילת השם, ובכל האותיות והמספרים ובסימנים \_ או \$ בהמשך השם.

שמות לא תקינים	שמות תקינים
3myVar מתחיל במספר	myVar3
my-var מכיל את התו - שאינו תקין	my_var
#myVar מכיל את התו # שאינו תקין	\$myVar
my var מכיל רווח	myVar

**שימוש לב:** אפשר להשתמש בעברית בהגדרת המשתנים, אבל מומלץ שלא לעשות את זה גם כיון שהקוד שלכם יהיה פחות קריא וגם כיון שהוא עלול להוביל להתנהגות מוזרה ולצורך. אתן לכם טיפ: בעולם התכנות אל תחשפו צורות כי יש מספיק מהן גם כך.

**שימוש לב 2:** בגרסאות קודמות של ג'אווהסקריפט השתמשו במילה השמורה var להגדרת משתנה. בתקנים החדשים כבר לא מקובל להשתמש ב-var כיון שלו שימוש בו יש השלכות שנדרן בהן בהמשך הספר. הוא נשאר איתנו בעיקר בשבייל תאמיות לאחר עבור קוד שנכתב בשנים עברו.

**תרגילים:**

צרו משתנה בשם המורכב לפחות משתי מילים, הדפיסו אותו בקונסולה וודאו שבהדפסה בקונסולה יופיעו המילים "I know JavaScript".

**פתרונות:**

```
let myVar;  
myVar = 'I know JavaScript';  
console.log(myVar);
```

**הסבר:**

יצירת המשתנה נעשית באמצעות המילה השמורה `let`. השמת הטקסט נעשית באמצעות הסימן `=`. שימושו לבש הטקסט מוקף בגרשיים. פקודת `console.log` שלמדנו בפרק הקודם משמשת להדפסת הטקסט - ובמקרה זהה המשתנה שהוא מקבלת.

**תרגיל:**

צרו משתנה בשם myVar והכניסו לתוכו את טקסט "me not know JavaScript". דרשו את הטקסט זהה בטקסט "I know JavaScript" והדפיסו אותו באמצעות הדרישה `console.log`

**פתרון:**

```
let myVar = 'me not know JavaScript';
myVar = 'I know JavaScript';
console.log(myVar);
```

**הסבר:**

יצרים את המשתנה `myVar` ומכניסים לתוכו את הטקסט "me not know JavaScript". ממש ברגע היצירה. לאחר מכן דורסים את הערך זהה באמצעות השמה נוספת. הדרישה של מה שיש במשתנה `myVar` נעשית באמצעות `console.log`.

**תרגיל:**

צרו משתנה בשם myVar2 והכניסו לתוכו את הטקסט "I am myVar 2". צרו משתנה נוסף בשם `myVar` והכניסו לתוכו את הטקסט "I am myVar". הדרישת `console.log` (בונוס: הדרישת שני המשתנים בקונסולה באמצעות פקודת `console.log` אחת)

**פתרון:**

```
let myVar = 'I am myVar';
let myVar2 = 'I am myVar 2';
console.log(myVar);
console.log(myVar2);
```

**הסבר:**

אין מינעה להגדר בינהו משתנים באותו סקריפט. הגדרתי את המשתנה באמצעות המילה השמורה `let`, שם המשתנה (הכולל מספר בסוף) והדפסה שלו ל-`console.log`.  
הסבר לתרגיל הבונוס: נסו לחפש באינטרנט איך להדפיס ל-`console` שני משתנים. הרבה פעמים יהיו דברים שלא תבינו בתרגול. במקומות לקרוא שוב ושוב את ההסבר, נסו לחפש תשובה בראשת. כך עושים מתכנתים מקצועיים: הם מוחפשים תשובה בראשת.

פרק 2

# טראנס

