מבוא לתכנות בסביבת אינטרנט

JavaScript (חלק 2 מתוך 3)

לפי תוכנית הלימודים של משרד החינוך

מהדורה שנייה ומעודכנת 2002

ספר זה בנוי על פי תוכנית הלימודים של משרד החינוך, האגף למדע וטכנולוגיות, מדעי המחשב ומגמת טכנולוגיות מידע. תוכנית זו, בהיקף של 90 שעות לימוד עיוניות ומעשיות, מיועדת לתלמידים הלומדים מדעי המחשב ברמה הרגילה שסיימו יסודות 1.

פנה לאתר האינטרנט של הספר כדי להוריד את קוד המקור של כל התוכניות וחומר נוסף הקשור לספר זה:

www.hod-ami.co.il/59355.html

עורך ראשי: **זהר עמיהוד** Microsoft Certified Professional

מרכז תוכנית beta: **צור ריכטר-לוין** Microsoft Certified Professional

עריכה ועיצוב: **טליה טופז**

עיצוב עטיפה: **שרון רז**

תודות:

תודה מיוחדת ל- **ד"ר אבי כהן**, המפמ"ר למדעי המחשב, שכתב את התוכנית והציג את הדרישות המקצועיות הפדגוגיות של הוראת הנושא.

אנו מודים למורים ולמורות שהעירו את הערותיהם והאירו את עינינו.

הערותיכם מאפשרות לנו להגיש לקהל הלומדים את המוצר הטוב ביותר האפשרי, ומאפשרות לתלמידים לעמוד בדרישות תוכנית הלימודים באופן הטוב ביותר.

תגובותיכם והערותיכם היו חשובות לנו מאוד ונעזרנו בהן רבות במהלך פיתוח חומר הלימוד בספר זה.

תודה מיוחדת למורים ולמורות (הרשימה מסודרת לפי א"ב):

איזבלה טבלין
שבח מופת, תל אביב

ישראל יוסף
תיכון חדש, תל אביב

אילן פרץ
מכללת אורט ע"ש יוסף חרמץ, ירושלים

נורית פינטל
תיכון בית רבקה

אילנה גורודצקי
מקיף ע"ש יצחק רגר, באר שבע

עופר סובול
העמק המערבי, קיבוץ יפעת

אלה לב
גימנסיה ריאלית ע"ש קררי, ראשון לציון

עמיחי פיגנבוים
אוניברסיטת בר-אילן, נוער שוחר מדע

ורד בראון
רנה קאסין, ירושלים

ראובן יגל
קריה חינוכית, מצפה רמון

טובי סטפ
עמל ב', פתח תקוה

שמעון אייבס
באר שבע

תודה גם ל: אייל בר-און, אלינור אורבך, אליק שפי, שחף בר-גל

שמות מסחריים

שמות המוצרים והשירותים המוזכרים בספר הינם שמות מסחריים רשומים של החברות שלהם. הוצאת הוד-עמי עושה כמיטב יכולת למסור מידע אודות השמות המסחריים המוזכרים בספר זה ולציין את שמות החברות, המוצרים והשירותים. שמות מסחריים רשומים (registered trademarks) המוזכרים בספר צוינו בהתאמה.

הודעה

ספר זה מיועד לתת מידע אודות מוצרים שונים. נעשו מאמצים רבים לגרום לכך שהספר יהיה שלם ואמין ככל שניתן, אך אין משתמעת מכך כל אחריות שהיא.
המידע ניתן "כמות שהוא" ("as is"). הוצאת הוד-עמי אינה אחראית כלפי יחיד או ארגון עבור כל אובדן או נזק אשר ייגרם, אם ייגרם, מהמידע שבספר זה, או מהדיסקט/תקליטור שעשוי להיות מצורף לו.

מבוא לתכנות בסביבת אינטרנט

לפי תוכנית הלימודים של משרד החינוך

מהדורה שנייה ומעודכנת 2002

זהר עמיהוד

ירון לייפנברג

תודה מקרב לב ל**זיו איילון**, מחבר הספרים **שפת C אל השיא** ו- **שפת C תרגול אל השיא** שנתן את רשותו להשתמש בתרגילים שחיבר.

הוצאת

הוד-עמי

ספרי מחשבים

© כל הזכויות שמורות

הוצאת הוד-עמי לספרי מחשבים בע"מ
ת.ד. 6108 הרצליה 46160
טלפון: 09-9564716 פקס: 09-9571582
www.hod-ami.co.il
info@hod-ami.co.il

אין להעתיק או לשדר בכל אמצעי שהוא ספר זה או קטעים ממנו בשום צורה ובשום אמצעי אלקטרוני או מכני, לרבות צילום והקלטה, אמצעי אחסון והפצת מידע, ללא אישור בכתב מאת ההוצאה, אלא לשם ציטוט קטעים קצרים בציון שם המקור.

לשם שטף הקריאה כתוב ספר זה בלשון זכר בלבד.
ספר זה מיועד לגברים ונשים כאחד
ואין בכוונתנו להפלות או לפגוע בציבור המשתמשים/ות.

מהדורה ראשונה 2001
מהדורה שנייה ומעודכנת 2002

All Rights Reserved
HOD-AMI Ltd.
P.O.B. 6108, Herzliya
ISRAEL, 2002

מסת"ב ISBN 965-361-320-0

תוכן עניינים מקוצר

הקדמה..... 1

פרק 1: מבוא ל- JavaScript 13

פרק 2: תחביר ושורות קוד ראשונות 25

פרק 3: מתחילים לכתוב..... 35

פרק 4: לולאות ובקרת זרימה 53

פרק 5: פונקציות..... 77

פרק 6: תמונות, מערכים ואנימציה..... 93

פרק 7: אובייקט המחרוזת..... 127

פרק 8: אובייקט התאריך 161

פרק 9: מסגרות..... 171

פרק 10: חלונות..... 181

סיכום..... 209

אינדקס..... 211

תוכן העניינים

הקדמה 1

פרק 1: מבוא ל- JavaScript 13

אובייקט - Object 15
תכונות - Properties 16
שיטות - Methods 16
HTML ואובייקטים 18
אובייקט בתוך אובייקט 19
Collections 19
פרויקט סיינפלד 23

פרק 2: תחביר ושורות קוד ראשונות 25

למתכנתי Pascal ו- C 25
כתיבת קוד JavaScript במסמך HTML 27
הסתרת קוד JavaScript 27
המחשב מברך 28
תחביר, תחביר, תחביר 31
פרויקט סיינפלד 34

פרק 3: מתחילים לכתוב 35

אובייקט המסמך - document 35
השיטה bgColor 36
שימוש בשיטה write() 37
שרשור 38
משתנים 39
שמות משתנים 39
הגדרת משתנה 42
ערכי משתנים 43
פעולות עם משתנים 45
אופרטורים 46
קביעת ערך 48
פרויקט סיינפלד 51

פרק 4: לולאות ובקרת זרימה 53

הלולאה while 53
הצבת ערך במשתנה 55
אופרטור מקוצר 56
הלולאה for 58
לולאה בתוך לולאה - לולאות מקוננות 59
הכנסת מספרים לטבלת HTML 61
התנאי if...else 62
הפקודה continue 67
הפקודה break 68
בדיקת ערך בוליאני 69
משפט switch 70
פרויקט סיינפלד 76

פרק 5: פונקציות 77

אירועים 77
רשימת אירועים 78
פונקציות 80
מיקום הפונקציות בתוכנית 81
קריאה לפונקציה 82
פונקציות מחזירות ערך 86
פרויקט סיינפלד 89

פרק 6: תמונות, מערכים ואנימציה 93

שמות אובייקטים 95
העברת נתונים לפונקציה 97
החלפת תמונות במעבר עכבר 100
אובייקט תמונה, טעינה מוקדמת 100
מערך 102
הגדרה והצבה 102
יתרונות המערך 105
שיטות הקשורות למערכים 106
אנימציה 108
השיטה setInterval() 110
מערכים רב-מימדיים 112
פרויקט סיינפלד 119
העשרה 122

פרק 7: אובייקט המחרוזת 127

אובייקט המחרוזת - String Object 127
טיפול בנתוני טופס 128
בדיקה ראשונית לטקסט שהוזן על ידי המשתמש 128
שינוי תוכן של שדה טקסט 133
שיטות נוספות של אובייקט מחרוזת 134
השיטה ()indexOf 134
השיטה ()lastIndexOf 136
השיטה ()split 137
השיטה ()substring 137
השיטה ()charAt 137
השיטות ()toUpperCase ו- ()toLowerCase 142
טיפול בנתוני טופס (המשך) 143
רשימה נגללת select 143
לחצני רדיו - Radio buttons 149
פרויקט סיינפלד 154
העשרה 159

פרק 8: אובייקט התאריך 161

שעון 163
הצגת שעון כחלק מדף HTML 166
תאריך, שעון וטופס 166
עוד על אובייקט התאריך 167
פרויקט סיינפלד 169

פרק 9: מסגרות 171

שמירת נתונים במסגרות סטטיות 172
כמה טיפים על מסגרות 178
פרויקט סיינפלד 179

פרק 10: חלונות 181

העברת מידע בין חלונות 185
תחרות מלכת היופי 2002 193
פרויקט סיינפלד 206

סיכום 209

אינדקס 211

הקדמה

מבוא לתכנות בסביבת אינטרנט

סדרת הספרים **מבוא לתכנות בסביבת אינטרנט** (שלושה חלקים) בנויה על פי תוכנית הלימודים של משרד החינוך, האגף למדע וטכנולוגיות, מדעי המחשב ומגמת טכנולוגיות מידע. תוכנית זו, בהיקף של 90 שעות לימוד עיוניות ומעשיות, מיועדת לתלמידים הלומדים מדעי המחשב ברמה הרגילה שסיימו יסודות 1.

סדרה זו מכסה את כל תכני הלימוד כפי שמופיעים בתוכנית הלימודים. תלמיד שילמד לפי ספר זה יוכל לעמוד בדרישות הקורס כפי שהוגדרו:

- הכרת עקרונות העברת המידע באינטרנט.
- הכרת המושגים הבסיסיים של תגיות HTML.
- הכרת המושגים הבסיסיים של תכנות תסריטים עם JavaScript.
- הכרת המונחים הבסיסיים של תכנות מבוסס אובייקטים.
- הכרת המושגים הבסיסיים של ASP.
- הכרת המושגים הבסיסיים של תכנות בצד הלקוח ובצד השרת.
- יכולת ליצור אתרים אינטראקטיביים מבוססי תסריטים בסיסיים.

פרקי **מבוא לתכנות בסביבת אינטרנט** מחולקים לשלושה ספרים:

הספר	כולל
מבוא לתכנות בסביבת אינטרנט, מבוא ו-HTML	1 פרק בנושא מבוא 9 פרקים בנושא HTML
מבוא לתכנות בסביבת אינטרנט, JavaScript	10 פרקים בנושא JavaScript
מבוא לתכנות בסביבת אינטרנט, ASP	10 פרקים בנושא ASP

חומר הלימוד המובא בשלושת הספרים הללו, כאמור, מכסה את התכנים, רשימת הפקודות, האובייקטים, האוספים, השיטות והמאפיינים הנדרשים על פי תוכנית הלימוד.

חומר עזר נוסף בנושאים אלה, ניתן למצוא בספרים הבאים בהוצאת **הוד-עמי**:

- **HTML 4 למפתחי אתרים באינטרנט**, זהר עמיהוד, הוצאת **הוד-עמי**.
- **JavaScript סדנת לימוד**, זהר עמיהוד, זיו אילון, הוצאת **הוד-עמי**.
- **JavaScript+CSS+DOM למפתחי אתרים באינטרנט**, זהר עמיהוד, הוצאת **הוד-עמי**.
- **JavaScript המדריך השלם**, R. Allen Wyke, הוצאת **הוד-עמי**.
- **ASP 3 המדריך השלם**, Stephen Walther, הוצאת **הוד-עמי**.
- **ASP 3 סדנת לימוד**, כולל Visual InterDev, רונן אלמוג, הוצאת **הוד-עמי**.

את הספרים ניתן להשיג:

ברשתות ובחנויות המובחרות,

ישירות בהוצאה בטלפון **1-700-7000-44**,

באתר האינטרנט של ההוצאה בכתובת: www.**hod-ami**.co.il.

כמו כן, מומלץ לעיין בספר:

- **עיצוב ממשק באינטרנט, Jackob Nielsen**, הוצאת **הוד-עמי.**

ספרות בנושא מערכת הפעלה Windows בהוצאת **הוד-עמי**:

Windows 95 תכל'ס, כ- 230 עמודים + תקליטון

Windows 95 הסדרה הידידותית, 528 עמודים

Windows 98 תכל'ס, כ- 240 עמודים + תקליטון

Windows 98 קוראים יודעים, כ- 592 עמודים + תקליטור

Windows 2000 קוראים יודעים, כ- 512 עמודים + תקליטור

Windows Me הסדרה הידידותית למתחילים, כ- 144 עמודים

Windows XP למשתמש בבית ובמשרד, כ- 294 עמודים

תוכן עניינים ופרק לדוגמה של כל ספר וספר ניתן למצוא באתר הוצאת הוד-עמי באינטרנט, בכתובת: **www.hod-ami.co.il**.

כיצד בנוי חומר הלימוד?

חומר הלימוד מורכב ממספר חלקים:

מבוא - הסבר קצר וענייני על התפתחות רשת האינטרנט ועל TCP/IP בצד מושגים שכיחים. חלק קצר זה גם יעסוק בנושא כתובת IP.

HTML - לימוד התגיות העיקריות בדף HTML להצגת תכנים (טקסט ותמונות), הכוללים: צבע, קישורים, טבלאות, טפסים ומסגרות.

JavaScript - לימוד שפת תסריט (script) דרך מעבר קצר על התחביר, אובייקטים, שיטות ותכונות (מאפיינים), טיפול בטפסים וחלונות.

ASP - עבודה מול שרת. הסבר עקרוני על דרך העברת המידע ברשת, דרך הכרת האובייקטים העיקריים, כולל עבודה עם קבצים.

דוגמאות

בספר זה תמצא כמה מאות דוגמאות. את קוד המקור של כל הדוגמאות בספר ניתן להוריד מאתר הוד-עמי, בכתובת:

www.**hod-ami**.co.il/**59355.html**

כל הדוגמאות של כל פרק נמצאות בתיקיה הנושאת את שם הפרק, כולל תמונות.

שים לב:

- קוד המקור המלא של כל הדוגמאות שבספר ניתן להורדה מהאתר.
- לעיתים, מובא בספר רק חלק מקוד המקור, כך שיהיה עליך לעיין בקוד המקור המלא שהורד מהאתר.
- כל הדוגמאות נבדקו והורצו בדפדפן Microsoft Internet Explorer בגרסאות 5.x ו- 6.x.

תרגול

במהלך הספר משולבים תרגילים לתרגול החומר הנלמד ולהטמעתו.

את קטעי הטקסט והתמונות הקשורים לתרגול ניתן להוריד מאתר הוד-עמי, בכתובת:

www.**hod-ami**.co.il/**59355.html**

כל קטעי הטקסט והתמונות בתרגול של כל פרק נמצאים בתיקיה הנושאת את שם הפרק.

פרויקט סיינפלד

במהלך הספר נבנה פרויקט מ-א' ועד ת'. זוהי דוגמה שלמה של פיתוח אתר לפי נושאי הלימוד של תוכנית הלימודים. את הפרויקט ניתן לבנות במקביל ללימוד הפרקים בכיתה. הפרויקט אינו בא במקום לימוד הנושאים שבפרק, אלא כתוספת חיזוק ותרגול.

מרבית החומרים לבניית הפרויקט (טקסטים ותמונות) ניתנים להורדה מאתר האינטרנט, בכתובת:

www.**hod-ami**.co.il/**59355.html**

מרבית קטעי הטקסט והתמונות הקשורים לפרויקט סיינפלד של כל פרק נמצאים בתיקיה הנושאת את שם הפרק.

תוכנות נדרשות

שלב לימוד	תוכנה נדרשת (דרישות מינימום)
מבוא	-
HTML	• עורך טקסט פשוט כמו **פנקס רשימות** (Notepad). התוכנה נמצאת כחלק מ-Windows. • דפדפן Microsoft Internet Explorer. רצוי בגירסה מעודכנת.
JavaScript	• עורך טקסט פשוט כמו **פנקס רשימות** (Notepad). התוכנה נמצאת כחלק מ-Windows. • דפדפן Microsoft Internet Explorer. רצוי בגירסה מעודכנת.
ASP	• עורך טקסט פשוט כמו **פנקס רשימות** (Notepad). התוכנה נמצאת כחלק מ-Windows. • דפדפן Microsoft Internet Explorer. רצוי בגירסה מעודכנת. • תוכנת Microsoft PWS או שרת IIS.

עורך טקסט מינימלי הוא עורך טקסט פשוט כמו **פנקס רשימות** (Notepad). ניתן גם להשתמש ב-Word או ב**כתבן** (WordPad), אבל יש לשמור את הקובץ כקובץ **טקסט** (txt) ולא כקובץ doc. באינטרנט ניתן למצוא עורכי טקסט משוכללים יותר המותאמים לעבודה עם HTML.

נחוץ דפדפן Microsoft Internet Explorer בגירסה מעודכנת. ספר זה נבדק בגרסאות 5.x ו- 6.x. פקודות ומאפיינים מסוימים, כמו גם אובייקטים ושיטות, עשויים שלא לפעול בגרסאות ישנות יותר (ובמיוחד גירסה 3 ומטה).

ASP 3 נתמכת במלואה רק בשרת האינטרנט של Microsoft בגירסה 5 (IIS 5). אבל, הפעלתו של שרת זה אפשרית רק למי שברשותו Windows NT 4 או Windows 2000 Server. מכיון שכך, חברת Microsoft שחררה מוצר הנקרא PWS (Personal Web Server) הניתן להתקנה ב- Windows 95/98. ב- Windows XP גרסת Professional יש כבר רכיב IIS, כך שלא נדרשת תוכנה נוספת.

היכן ניתן למצוא תוכנות אלו?

עורך טקסט

עורך טקסט (text editor) כמו תוכנת **Notepad** (הנקראת פנקס רשימות) שנמצאת בכל מערכת הפעלה Windows:

פתח את תפריט **התחל** (Start), **תוכניות** (Programs), **עזרים** (Accessories), **פנקס רשימות** (Notepad).

למי שמעוניין, יש מיגוון גדול של עורכי טקסט משוכללים יותר, freeware או shareware, אותם ניתן למצוא באתר www.tucows.co.il ובאלפי אתרים אחרים באינטרנט.

תוכנת Internet Explorer בגירסה מעודכנת

Windows 95/98/ME: ניתן לבצע שדרוג הגירסה באמצעות האינטרנט.

תוכנת Internet Explorer בגירסה מעודכנת נמצאת בתקליטור המצורף למרבית הספרים בהוצאת הוד-עמי (הגירסה מעודכנת ליום הדפסת התקליטור).

PWS

את תוכנת PWS ניתן למצוא בתקליטורים המצורפים לספרים הבאים בהוצאת **הוד-עמי**:

- **HTML 4 למפתחי אתרים באינטרנט**, מהדורה רביעית, זהר עמיהוד, הוצאת **הוד-עמי**.
- **JavaScript סדנת לימוד**, זהר עמיהוד, זיו אילון, הוצאת **הוד-עמי**.
- **JavaScript+CSS+DOM למפתחי אתרים באינטרנט**, זהר עמיהוד, הוצאת **הוד-עמי**.
- **JavaScript המדריך השלם**, R. Allen Wyke, הוצאת **הוד-עמי**.
- **ASP 3 המדריך השלם**, Stephen Walther, הוצאת **הוד-עמי**.
- **ASP 3 סדנת לימוד**, כולל Visual InterDev, רונן אלמוג, הוצאת **הוד-עמי**.

להלן טבלה המרכזת את נושא PWS במערכות ההפעלה השונות של Microsoft:

Windows 95	• התוכנה **לא** נמצאת בתקליטור ההתקנה של Windows 95. • נדרש לבצע עדכון ל- Winsock 2.0. • ניתן להתקין PWS.
Windows 98	• התוכנה נמצאת בתקליטור ההתקנה של Windows 98 בתיקיה \add-ons\pws • ניתן להתקין PWS.
Windows ME	• התוכנה לא נמצאת בתקליטור ההתקנה של Windows ME. • **לא** ניתן להתקין PWS במערכת הפעלה זו.
Windows XP	• במהדורת professional יש IIS **ולא** נדרש להתקין PWS. • במהדורת home **לא** ניתן להתקין PWS.
Windows 2000	• במהדורת Professional יש IIS **ולא** נדרש להתקין PWS. • במהדורת Server יש IIS **ולא** נדרש להתקין PWS.

אינדקס

בסוף הספר מצורף אינדקס בעברית ובאנגלית בו תוכל להיעזר כדי לחזור ולמצוא את מה שכבר למדת. חלק מהמושגים מופיעים רק באנגלית, ולכן מומלץ לעיין באינדקס באנגלית וגם באינדקס בעברית.

סימנים מוסכמים בספר

ספר זה מכיל מספר מוסכמות ואלמנטים שמטרתם לסייע לך במציאת מידע במהירות.

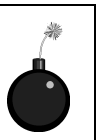
תרגיל

הערה

תחת הכותרת הערה תוכל למצוא מידע נוסף המרחיב את היריעה בנושא הנדון.

טיפ

תחת הכותרת טיפ תוכל למצוא מידע נוסף הנוגע לנושא הנדון במובן של קיצור דרך או טריק.

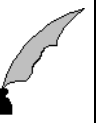


אזהרה!

האזהרות משמשות כתמרור המורה לך שקיימת אפשרות שתיתקל בבעיה, ומנחה אותך כיצד להימנע ממנה.

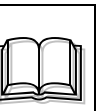
שאלה ותשובה!

תחת הכותרת שאלה ותשובה תמצא בעיקר מידע אודות שאלות נפוצות ופתרונן.



בעברית פשוטה!

נתקלת במונח או מושג חדש? תוכל למצוא הסבר תחת כותרת זו.



חומר עזר

הפנייה למקור/ות ידע.

תחביר, תחביר ועוד פעם תחביר

שפת HTML וכך גם JavaScript ו-VBScript כולן הינן "שפות חופשיות", במידה זו או אחרת. אם יש לך רקע של שפות תכנות כמו Pascal, C, ++C או Java בהן יש חוקים נוקשים של תחביר - כאן הדברים שונים. שפות התסריט JavaScript ו-VBScript הן שפות הנקראות Loosely Type Languages. מצד אחד זה "כייף" שניתן להשתמש במשתנה בלי להגדיר אותו, אבל מצד שני זה עשוי לבלבל ולהוביל לתוצאות "לא צפויות/שגויות", מכיון שמישהו אחר (במקרה זה הדפדפן, ה- script engine) עשה את העבודה עבורך ו... הוא יכול גם לטעות!

מעבר לכך, שפות כמו Pascal או C הן שפות "מבוססות" כבר שלושים וכמה שנים, לעומת שפות התסריט שקיימות לא יותר מכמה שנים. בשנותיהן הקצרות, עברו שפות התסריט הרבה מאוד גרסאות ועדיין "לא התבססו".

מכיון שרשת האינטרנט היא כל כך דינמית, שפות התסריט לא רק שצריכות להיות תואמות לאחור, הן לא מצליחות להתאים לתקנים המתפרסמים חדשות לבקרים: XML, ASP.NET, WAP, XHTML ועוד.

בספר זה נתעקש על כתיבת תחביר נכון. למרות שניתן לכתוב אחרת ונראה כאילו "ניתן לחסוך זמן ומאמץ", שווה לדבוק בכללי כתיבת קוד נכונים גם בשפות "חופשיות ומשוחררות", כמו JavaScript ו-VBScript. "החיסכון", לכאורה, יתבטל ברגע הראשון שתופיע בעיה כלשהי, והיא אכן תופיע!

World Wide Web היא רשת דינמית

יש לזכור תמיד, כי האינטרנט חיה ונושמת. אתרים חדשים נוספים לרשת בקצב של מאות ליום, בעוד שרבים אחרים גוועים ונעלמים. גם אתרים גדולים ומכובדים משנים את כתובתם מפעם לפעם.

הספר שבידך מכיל כתובות רבות ברשת. **כתובות אלו נבדקו בקפידה, אולם לא ניתן לעקוב אחר השינויים המתמידים של האינטרנט**. כך, שאם אתה מוצא עצמך מול כתובת כגון: Site was not found, או Document Not Found - אל ייאוש. בדוק את הכתובת שהקלדת, ונסה אותה בשינויים קלים. לעיתים קרובות תראה שינויים כאלה:

www.here.com/a_company_name/ ➔ www.a_company_name.com/

www.XYZ_company.com/ ➔ www.XYZ.com/

אך כאשר אתה תערוך שינויים באתר שתבנה בעתיד הקרוב, זכור להשאיר עקבות ברורים שינחו את קוראיך לכתובתך החדשה!

שים לב!

ספר זה מציג את תהליכי ההפעלה והגדרת התוכנות בסביבת מערכת ההפעלה Windows 95/98. Windows 9x מתאימה את המסכים לתצורת החומרה והתוכנה שמותקנים במחשב שלך. לכן, ייתכן שהמסכים שתראה במסך שבמחשב שלך יהיו שונים מעט מאלה שמתוארים בספר.

הערה חשובה!

הכתובות והמסכים בספר זה נאספו במהלך חודש ינואר/פברואר 2002. רשת האינטרנט מאוד דינמית, כפי שתיווכח במהלך העבודה, ולכן בהחלט ייתכן:

שמסכים שתראה על המסך שלך יהיו שונים מאלה המובאים בספר.

שכתובות לא תהיינה נכונות מאחר והדף הורד או שהמחשב נסגר.

הוצאת **הוד-עמי** עשתה, עושה ותעשה כל מאמץ להביא לפניך את החומר העדכני ביותר. לכן, השתדלנו שהכתובות המוזכרות בספר יהיו במחשבי שרת שימשיכו לתת שירותי אינטרנט עוד שנים רבות.

הערה נוספת חשובה!

הסיומות HTM ו-HTML זהות.

עברית

במי לבחור Internet Explorer או Netscape?

עם כל הכבוד ליוצרי Netscape שהיו חלוצים בתחומם - העברית היא לא הצד החזק של תוכנה זו. חברת Microsoft עם Internet Explorer עשתה עבודה טובה בכל הנוגע לעברית באינטרנט. לכן, מרבית הדוגמאות שתראה בספר זה הורצו על גרסאות 5.x ו- 6.x של תוכנת Microsoft Internet Explorer, מכיון שלכתוב דפי HTML בעברית זה קשה, אבל אם הם מוצגים בעזרת דפדפן Internet Explorer - קל לכתוב וקל לגלוש. שים לב, דף שנכתב בעברית לוגית לצפייה בדפדפן Internet Explorer לא ייראה טוב עם תוכנת Netscape. דע כי יש דרך לכתוב דף שיהיה טוב לצפייה עם Internet Explorer וגם עם Netscape אבל אתה תצטרך לעבוד, ולעבוד קשה. ובכלל לא בטוח שזה כדאי, מכיון שנכון לעכשיו למעלה מ- 88% מהגולשים בעולם משתמשים ב-IE.

עברית באינטרנט

מה לעשות, עברית אינה השפה הרשמית של האינטרנט. עברית כותבים מימין לשמאל כאשר עולם האינטרנט הפוך! לעברית יש אותיות (גופנים) שאין בשפה אחרת ו... חברות Microsoft ו-Netscape פיתחו את הדפדפנים שלהם שיציגו מסמכים ב... אנגלית.

אם כך, מה עושים כדי לראות ולעבוד בעברית באינטרנט? ובכן, עד כה פותחו שלוש גישות עיקריות להצגת העברית: Logical (על ידי חברת Microsoft), Semi-Logical ו-Visual (סנונית) וזוהי רק ההתחלה.

דפדפן Internet Explorer תומך בעברית במקור, ולכן אין צורך בהתקנות גופנים למיניהם. בעת התקנת דפדפן Microsoft Internet Explorer יש לבחור בתמיכה בהצגת השפה העברית.

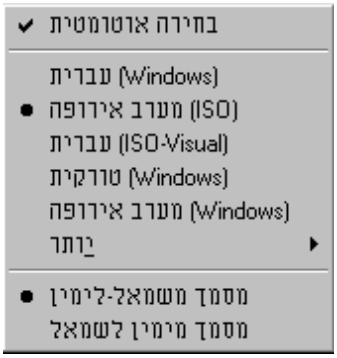
לא רואה עברית

לפעמים, בשל אופן כתיבת האתר (ולא ניכנס לפרטים הטכניים בנושא), Internet Explorer אינו מצליח לזהות את העברית באופן אוטומטי.

אם נכנסת לאתר והעברית בו אינה מוצגת כהלכה, אין בעיה!

1. לחץ לחיצה ימנית על נקודה ריקה כלשהי בדף בו אתה צופה.
2. בתפריט הקיצור הצבע על **קידוד** (Encoding).
3. בחר באפיון העברית הנדרש עבור האתר.

אינך רואה אפיון שפה כלשהו, או שהאפיון המבוקש אינו מופיע בתפריט? גם כאן יש פתרון: בתפריט המשנה של **קידוד** (Encoding) הצבע על **יותר** ותוכל לבחור אפיון אחר.



האפשרויות העומדות בפניך (בעברית) הן:
עברית (DOS),
עברית (ISO-Logical),
עברית (ISO-Visual),
עברית (Windows).

פעל בשיטת הניסוי והטעייה כדי לבחון את אפיון העברית המתאים לאתר בו אתה צופה.

בחלקו התחתון של התפריט מופיעות שתי אפשרויות: **מסמך משמאל-לימין** או **מסמך מימין-לשמאל**. אם גלשת לאתר בעברית שאינו מוצג נכון במסך (צמוד כולו לשמאל, במקום לימין), לחץ לחיצה ימנית במקום ריק כלשהו בדף בו אתה צופה, בתפריט הקיצור הצבע על **קידוד** וסמן בו את האפשרות **מסמך מימין-לשמאל**.

כתיבת עברית בעורך טקסט

עברית כותבים מימין לשמאל ואת כל השאר כותבים משמאל לימין. משפט זה לא בא ללמד אותך את שהינך כבר יודע, אלא לומר לך שאתה עשוי להיתקל בבעיות תצוגה. לדוגמה, אם אתה עובד עם **פנקס רשימות** (Notepad) ותרצה לרשום: "איזה יום יפה", זה ייראה כך:

תצוגה מימין לשמאל	תצוגה משמאל לימין
איזה יום יפה	איזה יום יפה

אבל אם תוסיף סימן קריאה (!) בסוף המשפט, משמאל למילה **יפה**, זה ייראה כך:

תצוגה מימין לשמאל	תצוגה משמאל לימין
איזה יום יפה!	!איזה יום יפה

אין פה טעות, יש פה בעיית תצוגה בלבד. כאשר תכתוב עברית מימין לשמאל המשפט ייראה בסדר, וכאשר תעבור לתצוגה שמאל לימין התצוגה תשתבש.

הערה

הסימן ↵ נמצא בשימוש רק בספר ופירושו, ששורה זו הינה המשך ישיר ורציף של השורה הקודמת. את ההמשך יש להקליד ברצף וללא לחיצת Enter. לחיצת Enter רק לאחר התו ; (נקודה-פסיק).

קבצי קוד המקור

לספר זה לא מצורף תקליטור.

את קוד המקור של כל הדוגמאות בספר ניתן להוריד מאתר הוד-עמי, בכתובת:

www.**hod-ami**.co.il/**59355.html**

שאלות, הערות, רעיונות

לא נחסך כל מאמץ להבטחת הדיוק של ספר זה וקבצי המקור הנלווים אליו. אם יש לך הערות, שאלות או רעיונות הנוגעים לספר זה של ההוצאה, אנא שלח אותם להוצאת הוד-עמי באחת השיטות הבאות:

דואר אלקטרוני: support@hod-ami.co.il, בשורת Subject ציין את המספר 59355

דואר רגיל:

הוצאת הוד-עמי לספרי מחשבים
ת.ד. 6108
הרצליה 46160

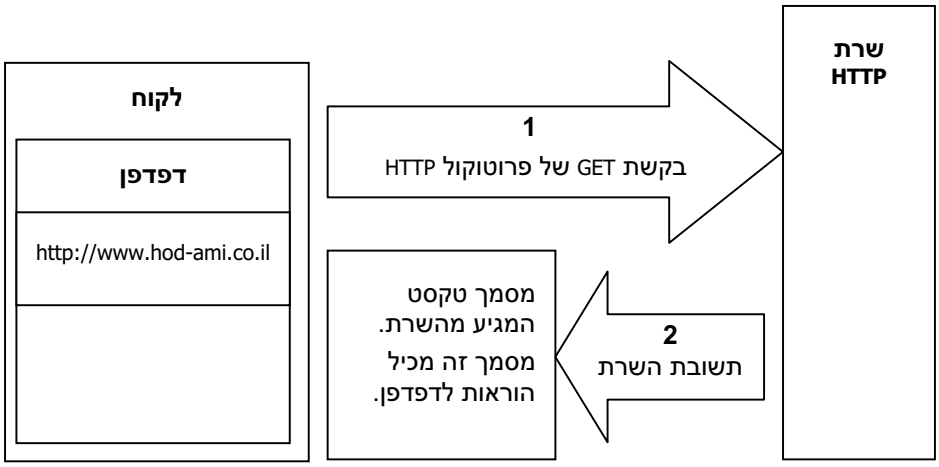
פרק 1

מבוא ל- JavaScript

כדי להבין את שפת JavaScript, יש להבין את הסביבה בה אנו מפתחים - סביבת HTTP.

בסביבת עבודה זו ישנם **שרתי HTTP** ו**לקוחות**. השרתים הם מחשבים המחוברים לרשת האינטרנט 24 שעות ביממה וממתינים להעניק שירות ללקוחות הפונים אליהם. כאשר אתה מקליד בשורת הכתובת בדפדפן את הטקסט הזה: http://www.hod-ami.co.il או רק www.hod-ami.co.il, המחשב שלך פונה כלקוח (client) אל מחשב השרת (server) של הוצאת הוד-עמי ומבקש ממנו בקשת HTTP. הבקשה היא בקשת GET כלומר, המחשב שלנו, הלקוח, מבקש מהשרת לקבל דבר מה.

השרת שולח קובץ טקסט אל המחשב שלך ובזאת ניתק הקשר. תהליך פשוט זה הינו הבסיס לכל הטכנולוגיה. התרשים הבא מתאר את תהליך הבקשה והתשובה.



תרשים 1.1

מסמך הטקסט שמגיע מהשרת מכיל למעשה הוראות לדפדפן. הדפדפן מאחסן את ההוראות בזיכרון הזמני שלו ומבצע אותן. כפי שבוודאי ניחשת, מסמך הטקסט המגיע מהשרת הוא למעשה **מסמך HTML**.

אם כך, הדפדפן מקבל מסמך HTML ומתחיל לבצע את ההוראות החל מהפינה השמאלית העליונה, תוך שהוא קורא את השורות משמאל לימין.

הדפדפן מבצע בקלות הוראות רבות, כמו למשל: <body bgcolor="teal">, מכיון שפקודות HTML הפשוטות אינן דורשות חישובים או הקצאת מקום בזיכרון. כיום, דפדפנים שונים יודעים לבצע פקודות נוספות הנבדלות מפקודות HTML ברמת התחכום שלהן והכוללות: הקצאת משתנים, יצירת לולאות, מערכים ועוד. כל ההוראות האלו מתבצעות על ידי הדפדפן ואינן מפעילות את השרת בפעולות חישוב כלשהן. סוג זה של תכנות נקרא **תכנות בצד הלקוח** (Client side programming).

שפת JavaScript היא אחת מבין שתי שפות תסריט (script) בצד הלקוח. התסריט נרשם כחלק ממסמך HTML ומתבצע על גבי מחשב הלקוח. שפת צד לקוח נוספת היא VBScript של חברת Microsoft, אך היא מוכרת לדפדפן Microsoft Internet Explorer בלבד, בשעה ש-JavaScript מוכרת גם לדפדפן Netscape.

JavaScript היא **שפה מבוססת אובייקטים** (Object Based), השונה למשל משפת C++ שהיא **שפה מוכוונת עצמים** (Object Oriented). הדיון בהבדל בין Object Based ל- Object Oriented הוא מחוץ למסגרת ספר זה. תחילה, נסביר את תפיסת השימוש באובייקטים (יש הקוראים לאובייקטים בשם עצמים).

חומר עזר

נושא שפות התסריט, התחביר שלהן וההבדלים ביניהם נסקר בהרחבה בספר **JavaScript+CSS+DOM למפתחי אתרים באינטרנט**, בהוצאת **הוד-עמי**.

בבניית אלגוריתם, שהינו תרשים פעולה עבור תוכנית מחשב, עליך להנחות את המחשב בכל צעד. למשל, אם תרצה לתאר נסיעה במכונית, תצטרך לתאר למחשב את המכונית על כל רכיביה ולהסביר את תהליך ההצתה ופעולת הבוכנות, את פעולת התמסורות וסיבוב הגלגלים. לאחר שתעשה זאת, יהיה עליך לחזור על התיאור עבור כל נסיעה או עבור כל מכונית אחרת. בדרך זו תיאלץ לכתוב שורות קוד רבות. ניתן לקצר תהליך זה על ידי שימוש באובייקטים.

שאלה 1.1

מה המשמעות של תכנות בצד הלקוח (client side programming) מבחינת הדפדפן ומבחינת המשתמש? הצג דוגמה והסבר.

אובייקט - Object

אובייקט הוא תפיסה בה נוהגים לתאר את הסובב אותנו (למשל: מכונית, בית וכדומה) כדי שהמחשב יבין ויהיה קל לכתוב קוד. אפשר להתייחס לאובייקטים כאל **מכולות** (containers) של מידע. הבית מכיל מספר חלקים: סלון, מטבח, חדר שינה. אם ניקח את הסלון, לדוגמה, גם הוא מכיל מספר חלקים: טלוויזיה, כורסה, שולחן, שטיח שכל אחד מהם מורכב מחלקים נוספים עד לבורג האחרון. אם נתייחס למכונית, נראה שגם היא מורכבת מחלקים (מכלולים) בתוכה: הגה, מנוע, כסאות שכל אחד מהם הוא מכלול של חלקים אחרים.

אתה לא רק מכיר אובייקטים אלה אלא גם עובד איתם, רק שאף פעם לא קראת להם... אובייקטים. למשל, מערכת הקבצים ב-Windows היא אובייקט. אם נתייחס לדיסק C בתור אובייקט, אז הוא מכיל תיקיות (כמו Windows, My Documents ועוד). בתיקיה Windows נמצאות תיקיות נוספות כמו System, Media וכדומה, ואכן הצורה ההיררכית הזאת היא הדרך בה מתואר אובייקט למחשב.

ניתן להסתכל על אובייקטים הנמצאים בתוך אובייקטים כמו קופסה הנמצאת בתוך קופסה בתוך קופסה. התיקיה Windows היא אובייקט בתוך אובייקט הנקרא "דיסק C".

ב-JavaScript משתמשים ב**תחביר הנקודה** (dot syntax), המציין שייכות של משהו למכלול ברמה גבוהה יותר. אובייקט המכונית ייקרא בשם CAR. הגה, שגם הוא אובייקט, שהוא חלק ממכונית יירשם CAR.WHEEL ומנוע, שגם הוא אובייקט, של המכונית יירשם CAR.ENGINE.

תכונות - Properties

אובייקט המכונית נקרא בשם CAR (כמובן שהוא יכול להיקרא בכל שם אחר, אולי mooki?). למכונית יש **תכונות** (properties): צבע, מספר דלתות, רוחב, אורך, מהירות ועוד. כל המידע הזה נמצא מוגדר באובייקט, כך שבפעם הבאה שיהיה צורך לתאר למחשב שלך מכונית, תעשה שימוש באובייקט CAR על ידי ציון השם בלבד - CAR.

התכונה COLOR מתארת את צבע המכונית. ב-JavaScript משתמשים ב**תחביר הנקודה** (dot syntax), המציין שייכות של משהו למכלול ברמה גבוהה יותר. כלומר, כדי להתייחס לתכונת "צבע" של "מכונית", יש לכתוב CAR.COLOR. באותו אופן תוכל להתייחס לתכונות של אובייקטים ברמה נמוכה יותר (נמוכה יותר מרמת האובייקט CAR). כדי להתייחס לצבע ההגה יש לכתוב CAR.WHEEL.COLOR, וכדי להתייחס לצבע הרדיו יש לכתוב CAR.RADIO.COLOR. ולבסוף, כדי להתייחס לצבע כפתור השמע ברדיו יש לכתוב CAR.RADIO.VOLUME.COLOR.

כפי שנאמר, מערכת הקבצים במחשב היא אובייקט שיש לו תכונות. אם ניקח את דיסק C בתור אובייקט, אז יש לו תכונה המתארת את שמו. כדי להתייחס לתכונה זו נכתוב c.name. כל תיקיה שבדיסק C (כלומר, באובייקט C) היא גם אובייקט ונתייחס אליה כך: C.FOLDER. גם לתיקיה יש תכונות כמו name ונתייחס אליהן כך: C.FOLDER.NAME. גם לקובץ יש תכונות כמו ReadOnly, Hidden, Size וכדומה, ונתייחס אליהן כך: C.FOLDER.FILENAME.SIZE.

שיטות - Methods

לאובייקטים יש גם **שיטות** (methods). שיטות הן פעולות השייכות לאובייקט. פעולה השייכת לאובייקט המכונית (CAR) היא לדוגמה, נסיעה, אשר תצוין בשם DRIVE. כלומר, כשתרצה לגרום למכונית לנסוע, יהיה עליך לכתוב CAR.DRIVE().

כדי להפעיל במדויק את הפעולות המוגדרות על ידי שיטות, צריך לציין פרמטרים שיצוינו בין הסוגריים. כדי להפעיל נסיעה קדימה יש לרשום CAR.DRIVE() ללא כל ציון פרמטרים, אבל אפשר לרשום ערך, למשל CAR.DRIVE(FAST). כדי לעצור את המכונית, נשתמש בשיטה STOP() ונפעיל אותה כך: CAR.STOP().

באותו אופן תוכל להפעיל שיטות על אובייקטים ברמה נמוכה יותר. כדי לשנות את עוצמת הצליל ברדיו שבמכונית, יהיה עליך להפעיל את השיטה CAR.RADIO.VOLUME(LOW). כדי לפתוח את החלון הימני הקדמי, רשום CAR.WINDOWS.LEFTFRONT.OPEN().

נחזור למערכת הקבצים שהיא אובייקט. יש שיטות ברמת הדיסק כמו format ויש שיטות ברמת התיקיה: העתקת תיקיה, מחיקת תיקיה, שינוי שם תיקיה. יש גם שיטות ברמת הקובץ: מחק קובץ, שנה שם קובץ וכדומה.

כעת, משהוגדר אובייקט המכונית ואף הוגדרו השיטות והתכונות שלו, יהיה ניתן לפנות אל האובייקט בכל זמן. הדבר היפה בתפיסת האובייקטים הוא, שצריך להגדיר את האובייקט פעם אחת בלבד. לדוגמה, מספיק שפעם אחת אתה תגדיר אובייקט בשם CAR על כל שיטותיו ותכונותיו. מעכשיו, תוכל לעשות בו שימוש לא רק בתוכנית הנוכחית, אלא בכל תוכנית הזקוקה לאותו אובייקט מסוג CAR. יהיה לנו קל מאוד לשנות את התכונות של האובייקט ולהפעיל את שיטותיו ויהיה לאחרים קל לגשת לאובייקט שיצרת, כי הכל מובנה וברור.

◆ שאלה 1.2

תאר עצם (בית, אופניים, כיור, טלוויזיה, טלפון) כאובייקט. ציין לפחות מספר אובייקטים, שיטות ותכונות (עדיין בלי לשייך שיטות ו/או תכונות לאובייקט כלשהו).

◆ שאלה 1.3

מערכת מיני סטריאו היא אובייקט. ציין לפחות 5 שיטות ו-5 תכונות לאובייקט זה.

HTML ואובייקטים

מסמך HTML הוא אובייקט, אבל הוא לא האובייקט "הגבוה" ביותר (בראש ההיררכיה). מסמך HTML (אובייקט בשם document) נמצא בתוך חלון, לכן הוא נמצא בתוך אובייקט החלון בשם window ולכן נרשום אותו כך: window.document. כמו כל אובייקט גם לאובייקט window יש שיטות ותכונות, למשל: open(), close() אותן נכתוב window.open() ו-window.close(). במילים אחרות, אם תרצה שתסריט JavaScript יסגור את החלון הנוכחי יהיה עליך לכתוב: window.close(). ואם תרצה להדפיס את הדף מתוך התסריט? window.print().

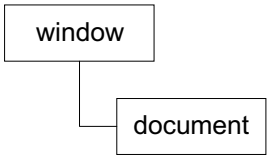
לאובייקט window יש גם תכונות כמו innerWidth, innerHeight, statusbar וכדומה. את ערכי התכונות תוכל לקרוא במהלך התסריט שתכתוב ו/או לתת להן ערכים. למשל, תוכל לקרוא את ערך התכונה innerWidth של החלון, אותה כותבים window.innerWidth, כדי לדעת מהו רוחב החלון ואולי להציג הודעה למשתמש שכדאי לצפות באתר זה בחלון ברוחב מקסימלי. אפשר, אם תרצה, לקבוע את גודל החלון, רוחב ואורך, עם התכונה innerWidth. למשל: window.innerWidth=300px יקבע את רוחב החלון ל-300 נקודות (pixels).

שאלה ותשובה!

האם צבע הרקע הוא תכונה של אובייקט החלון window?

לא. צבע הרקע אינו תכונה של אובייקט החלון (window) אלא של אובייקט המסמך (document).

לאובייקט של מסמך HTML קוראים בשם document, כך שכדי לתאר את מודל מסמך HTML ניתן לכתוב: window.document. וגם לאובייקט המסמך יש שיטות ותכונות משלו.

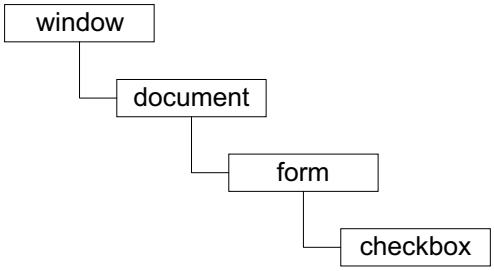


לאובייקט document יש שיטות משלו. למשל, יש לו שיטה close() אותה כותבים window.document.close(). זו אינה השיטה window.close() והיא בכלל לא דומה לה. משתמשים בשיטה window.document.close() כדי להפסיק רצף מידע המיועד למסך וטרם הוצג.

לאובייקט document יש גם תכונות משלו. למשל, תכונה המתארת/קובעת את צבע הרקע של המסמך - window.document.bgColor.

אובייקט בתוך אובייקט

אובייקט המסמך (document) נמצא בתוך אובייקט החלון (window). באובייקט המסמך נמצאים אובייקטים נוספים. תוכן הדף קובע את האובייקטים שבתוך אובייקט document. אובייקט document נוצר בעקבות הימצאות התגית <body>. לדוגמה, אם הוגדר טופס בעזרת התגית <form>, אז נוצר אובייקט form בתוך אובייקט document. אם הוגדר שדה בטופס בעזרת התגית <input type="checkbox" />, אז נוצר אובייקט checkbox תחת אובייקט form שתחת אובייקט document והרי לך היררכיה.



לכל אובייקט יש שיטות ותכונות משלו ובמרבית המקרים יש לו גם יכולת להכיל בתוכו אובייקטים נוספים.

הערה

המונחים **Function** ו-**Method** תורגמו למילים **פונקציה** עבור Function ו**שיטה** עבור Method. באופן בסיסי, שיטה היא פונקציה של אובייקט. פונקציה יכולה להיות פונקציה שאתה כתבת או פונקציה built-in של המערכת. כאשר פונקציה שייכת לאובייקט היא נקראת **שיטה** (method).

המונחים **Attribute** ו-**Property** תורגמו למילים **מאפיין** עבור Attribute ו**תכונה** עבור Property. במאפיין נשתמש כחלק מתגית HTML, למשל <body bgcolor="blue">, שהתכונה המקבילה לו במודל האובייקטים היא bgColor של האובייקט document הנכתבת כך : document.bgColor.

Collections

רעיון חדש יחסית בתחום פיתוח הקוד לדפדפנים באינטרנט - ה**אוספים** (collections). אוספים הם קבוצות של אובייקטים המקובצים יחדיו במטרה לחשוף את כל מרכיביו של האובייקט לתסריטים. האוסף הנמצא בראש ההיררכיה הוא האוסף all של האובייקט document. לכל אובייקט בתוך המסמך יש אוסף משלו. ישנם איפוא אוסף **תמונות** (images), אוסף **עוגנים** (anchors), **יישומונים** (applets), **אובייקטים** (objects) ועוד.

טבלה 1.1: האוספים ב-IE (רשימה חלקית מתוך 138 אוספים קיימים)

all	cells	frames	plugins
anchors	childNodes	images	rows
applets	children	imports	rules
areas	controlRange	links	scripts
attributes	elements	mimeTypes	styleSheets
behaviorUrns	embeds	namespaces	tBodies
bookmarks	filters	options	TextRange
boundElements	forms	pages	TextRectangle

אם כן, מודל האובייקטים מאפשר חשיפה של האובייקט לתסריט במספר צורות. שפת JavaScript, שפת תסריט, מאפשרת גישה למודל האובייקטים הנקרא **DOM** (**D**ocument **O**bject **M**odel) של IE. כלומר, בעזרתה ניתן להפעיל שיטות, לגשת ולקרוא ו/או לשנות ערכי תכונות של אובייקטים ועוד.

חומר עזר

מידע מקיף על מודל האובייקטים תוכל למצוא בספר **JavaScript+CSS+DOM למפתחי אתרים באינטרנט** בהוצאת **הוד-עמי**.

שאלה 1.4

תאר 3 אובייקטי משנה (תת-אובייקטים) לאובייקט מערכת מיני-סטריאו. ציין לפחות 3 שיטות ו-4 תכונות לכל אובייקט.

שאלה 1.5

סמן אובייקט, שיטה או תכונה. סמן + במשבצת המתאימה:

	אובייקט	שיטה	תכונה
אקולייזר 5 ערוצים			
הוצא קסטה			
זמן שעבר מתחילת הרצועה			
טיונר דיגיטלי AM/FM			
טייפ דאבל קסט			
כבה מכשיר			
לחיצה על לחצן Play בשלט			
מספר רצועה שמתנגנן עכשיו			
מצב אוזניות			
מצב לחצן Play בטייפ			
משקל			
מתח הפעלה חשמלי			
הנמך volume			
סריקת תחנות רדיו			
פתח מכסה קומפקט דיסק			
קומפקט דיסק			
רמקולים			
שלט רחוק			
שמור תחנת רדיו בזיכרון			

שאלה 1.6

חשבון עובר ושב (עו"ש) הינו אובייקט. ציין אובייקטים, שיטות ותכונות לאובייקט זה.

פרויקט סיינפלד

ראה את המשך הפרויקט בפרק 5.

פרק 2

תחביר ושורות קוד ראשונות

למתכנתי Pascal ו- C

שפת התכנות JavaScript היא שפת תכנות ככל שפת תכנות המוכרת לך: Pascal, C או C++. כמו בכל שפת תכנות יש בה: משפטים (קלט, פלט, השמה, תנאי, בקרה), משתנים, אופרטורים, מערכים ופונקציות. לכן, שפת JavaScript תיראה לך מוכרת: יש בה if, do..while וגם switch. אז אולי פה ושם יש שוני בתחביר ויש לשים לב, אבל באופן עקרוני התחביר ייראה לך מוכר.

אבל, JavaScript גם שונה משפות תכנות אחרות כי היא נולדה ברקע שונה. להלן טבלה המסכמת את ההבדלים העיקריים (ויש עוד) בין שפת JavaScript לשפת C:

	שפת JavaScript (נכון לשנת 2001/2)	**שפת C** (נכון לשנת 1983)
סביבת עבודה	רשת, אינטרנט Penium 4, 1500MHz, 128MB זיכרון פנימי	DEC-PDP-7 8088, 4.77MHz 64Kb זיכרון פנימי
צורך	עבודה עם טפסי HTML	חישובים מתמטיים
הכרזה	1995	1971

בעוד שפת C חוגגת כבר יום הולדת שלושים, לשפת JavaScript מלאו בקושי 7 שנים. כאשר ריצ'י פיתח את שפת C בשנת 1971 משפות תכנות אחרות (ויותר ותיקות), המחשב החזק שהיה ברשותו נקרא DEC-PDP-7 והמחשב האישי הראשון של IBM, שהוכרז בשנת 1983, היה חזק/מהיר/יעיל ממנו פי כמה וכמה (כן, המחשב עם מעבד במהירות 4.77Mhz היה יותר מהיר מה-PDP-7).

בסביבת עבודה (כזו לחוצה מבחינת נפח זיכרון ויכולת מעבד) בה כל סיבית (bit) וכל בית (byte) היו חשובים, היתה משמעות למספר הבתים (bytes) שתופס משתנה. לכן, בשפת C תמצא משתנים מספריים כמו: int, float, short, long, double, long double, ואילו בשפת JavaScript כל מספר מטופל כנקודה צפה (floating-point) שהוא הדבר הכי בזבזני מבחינת הזיכרון.

כשריצ'י פיתח את שפת C, בתחילת שנות השבעים, הוא חשב על פיתוח שפת תכנות שבעזרתה ניתן יהיה לכתוב תוכניות שישלחו טיל לחלל, יחשבו משוואות מתמטיות מורכבות, יעסקו בניתוח סקרים עם הרבה חישובים מתחום הסטטיסטיקה וכדומה. כשמתכני חברת Netscape ו-SUN פיתחו את JavaScript, באמצע שנות התשעים, הם חשבו על שפה שתעבוד במסגרת הדפדפן ותוכל לעבד נתונים שנשלחו מהמשתמש. במילים אחרות, הם חשבו על שפת תכנות הפועלת בשרת אינטרנט שתוכל לעבד נתונים שנשלחו מטפסי HTML שמילא הגולש באתר. אלו הן שתי נקודות מוצא שונות לחלוטין, שהובילו לכיווני פיתוח שונים.

הבדל חשוב נוסף בין שפת תכנות כמו C לשפת JavaScript הוא תהליך הפעלתה. כדי להפעיל תוכנית הכתובה בשפת C, עליה לעבור הידור (compilation). המחשב מבין 0 או 1 ולמרות שאתה כותב לו, בשפת C, את המשפט:

```
if (num % div != 0) continue;
```

אין לו מה לעשות עם זה כי הוא מבין רק.. 0 או 1. המהדר (compiler) לוקח את התוכנית שכתבת בשפת C ומתרגם אותה לקבצים בינאריים (0/1), המהווים תוכניות מחשב הניתנות להפעלה (excutables). אם תנסה להציץ בתוכנו של קובץ בינארי, תראה קוד מכונה (אם בכלל תוכל לראות משהו) ולא את המשפטים המקוריים אותם כתבת. בשפת JavaScript אין מהדר. הדפדפן מפענח (ויש הבדל בין הידור לפיענוח) את התוכנית שכתבת on-the-fly. כלומר, במהלך ביצוע הקוד. תהליך זה זכה לשם interperter או בעברית מפרש. הדפדפן לא יוצר קובץ exe, אלא לוקח שורה, מתרגם, מפעיל, עובר לשורת הקוד הבאה וחוזר חלילה.

JavaScript היא שפה מתירנית (Loosely Typed Language) מבחינת הכתיבה. כלומר, אין בה חוקים מחמירים בנוגע לדרכים בהן משתנים (אובייקטים) נשמרים בזיכרון. אין צורך להכריז על אובייקטים אלה בזיכרון, המכונים גם משתנים, כמכילים סוג מסוים של נתונים. JavaScript תגלה באיזה סוג נתונים מדובר בהתאם לשימוש שתעשה בהם. שפות תכנות רבות, כמו C, דורשות ציון של סוג המשתנה בעת ההכרזה עליו, זאת מפני שההבחנה בין משתנה מסוג float (מספרים לא-שלמים עם נקודה עשרונית להם מוקצים 4 bytes בזיכרון) לבין משתנה מסוג double (מספר לא-שלם לו מוקצים 8 bytes בזיכרון), או בין סוגים נוספים של נתונים, היא חשובה. בעבודה עם JavaScript אתה יכול לשנות את סוג המשתנה תוך כדי פעולת התוכנית, בלי לציין מהו סוג הנתונים החדש. במילים אחרות, שפת JavaScript מסוגלת לקבוע באילו סוגי נתונים מדובר מתוך הקשר.

כתיבת קוד JavaScript במסמך HTML

הדפדפן מפרש את קוד HTML. שפת HTML מתייחסת לכל תו שאינו תחום בין סוגריים זוויתיים (< >) כאל טקסט, ומציגה אותו על המסך.

כדי להימנע מהצגת קוד JavaScript (שתחבירו אינו כולל שימוש בסוגריים זוויתיים, < >), יש להורות לדפדפן על המעבר מ-HTML ל-JavaScript. במילים אחרות, עלינו להגדיר לדפדפן תחום מסוים בו תיכתב שפת JavaScript על ידי שימוש בתגית <script>, המבהירה כי מעתה ועד לתגית הסגירה </script> ייכתב קוד JavaScript. מכיון שמלבד JavaScript קיימות גם VBScript וגם JScript, יש להבדיל ביניהן על ידי הוספת המאפיין language, כפי שמוצג בדוגמה הבאה.

```
<script language="javascript">
JavaScript         כאן ייכתב קוד
</script>
<script language="vbscript">
VBScript          כאן ייכתב קוד
</script>
```

אם לא תציין ערך למאפיין language בתגית <script>, דפדפן IE יתייחס לשפת התסריט JScript שזוהי הגירסה המיקרוסופטית ל...JavaScript.

חומר עזר

מידע מקיף על **JScript** תוכל למצוא בספר **JavaScript+CSS+DOM למפתחי אתרים באינטרנט** בהוצאת **הוד-עמי**.

הסתרת קוד JavaScript

כדי למנוע מצב בו דפדפן מיושן או דפדפן שאינו מכיר את התגית **<script>** ימשיך להציג על המסך את קוד JavaScript שבדף HTML, רצוי לתחום את הקוד במסגרת הערת HTML. דפדפן המבין את התגית <script> לא יתייחס להערת HTML, מכיון שמבחינתו הוא יודע שקיים שם קוד JavaScript. דפדפן שאינו מכיר את התגית <script> יתעלם לחלוטין מהקוד העטוף בהערה, כפי שמוצג בדוגמה הבאה.

```
<script language="javascript">
<!--
Here comes JavaScript code
//-->
</script>
```

הוספת תגית ההערה חשובה בעת פרסום האתר באינטרנט, אולם הוספה זו אינה דרושה בעבודה וגם לא בעת התרגול בספר זה. לכן, אין צורך להוסיף הערה בכל תרגיל.

בעברית פשוטה!

אנו כותבים JavaScript בין תגית הפתיחה <script language="javascript"> לבין תגית הסגירה </script>.

המחשב מברך

דוגמה **2-01.html** : העתק את הקוד הבא ושמור אותו כמסמך HTML.

```
<html>
  <head>
<script language="javascript">

   window.alert("Welcome to Jamaica man!") ;

</script>
    <title>
    </title>
  </head>
<body>
</body>
</html>
```

בדוגמה זו השתמשנו באחת השיטות (method) הפופולריות של השפה הנקראת alert() והשייכת לאובייקט window. השיטה window.alert() מציגה חלון הודעה למשתמש. בהמשך הספר נעסוק בשיטה זו בהרחבה, ואף נשתמש בה לבדיקת הקוד על ידי הצגת הודעות במהלך ריצת התוכניות.

פתיחת דף זה על ידי הדפדפן תגרום להופעת ההודעה הבאה:

תרשים 2.1 : חלון הודעה הנוצר על ידי window.alert().

בעברית פשוטה!

השיטה **window.alert()** מציגה חלון הודעה למשתמש, ועוצרת את ריצת התוכנית עד שהמשתמש לוחץ על לחצן **אישור**.

ועכשיו בעברית. הפעל את קובץ **2-02.html** שאינו שונה מקובץ **2-01.html**, פרט לנוסח ההודעה. השיטה window.alert() משמשת בדרך כלל למתן הודעות קצרות למשתמש. אתה, כמפתח אתר באינטרנט, תוכל להיעזר בה גם להצגת ערכי משתנים לצורך בדיקה.

בעיקרון, ניתן לכתוב את קוד JavaScript בכל מקום במסמך HTML.
הקוד יתבצע באופן יחסי לשאר התגיות (יש לתחום את הקוד בין תגית הפתיחה **<script language="javascript">** לבין תגית הסיום **</script>**). עם זאת, בשלב מאוחר יותר, כאשר נדון בפונקציות, נכתוב את הקוד בתוך הכותרת, <head>, של המסמך.

הערה

בעת גישה לשיטות ולתכונות של אובייקט window לא חובה לציין את שמו, כך שאת השיטה alert() של אובייקט window אפשר להפעיל כך:

```
window.alert("Houston, we have a problem!") ;
```

וגם כך:

```
alert("Houston, we have a problem!") ;
```

את האובייקט window לא חייבים לציין. אני מציע כן לכתוב את שם האובייקט כדי למנוע טעויות בהמשך.

בטיסה לירח של החללית אפולו 13 (שנת 1970), לאחר שהאסטרונאוטים הבינו שיש להם בעיה, נשמע בקשר:

```
Houston, we have a problem!
Houston, do you read?
```

ומאז מסמל המשפט הזה סימן מצוקה.

שאלה 2.1

האם חובה לציין את המאפיין language בתגית <script>? אם לא מציינים מאפיין זה, מהי שפת התסריט?

שאלה 2.2

האם את התגית <script> ניתן לכתוב **רק** בין התגית <head> ל- </head>?

תחביר, תחביר, תחביר

- תחביר ה-DOM על כל האובייקטים, האוספים, השיטות והתכונות, רגיש לאותיות רישיות (גדולות). השיטה ()window.alert תעבוד, ואם תכתוב ()window.Alert תקבל הודעת שגיאה בנוסח:

```
Object doesn't support this property or method
```

- שפת JavaScript רגישה לאותיות רישיות (גדולות) ולאותיות רגילות (קטנות). לכן, רצוי ומומלץ להקפיד על שימוש באותיות רגילות, אלא אם כן מצוין אחרת. כותבים if ולא IF, for ולא For וכדומה.
- בסוף משפט/פקודה בשפת JavaScript לא חובה לשים ; (נקודה-פסיק), אבל מטעמים של נוהלי תכנות תקינים אנו נקפיד על ; בסיום כל משפט.

שאלה 2.3

האם בסוף משפט בשפת JavaScript חובה לציין ; (נקודה-פסיק)?

- פעמים רבות דרושים גרשיים. ניתן להשתמש בגרש בודד (') או בגרשיים ("), אך יש להקפיד לא לבלבל בין השניים. גם אסור להשתמש באותו הסוג זה בתוך זה. בדוגמה שלפניך מוצג מה **נכון** ולמה **לא נכון**.

לא נכון:

```
alert("He was known as "The Snake" all through the land!") ;
document.write("<body bgcolor="red"
↳text="yellow"><h1>Hello</h1>") ;
```

נכון:

```
alert("He was known as 'The Snake' all through the land!") ;
alert("He was known as " + "'The Snake'" + " all through the land!") ;
document.write("<body bgcolor='red'
↳text='yellow'><h1>Hello</h1>") ;
```

שאלה 2.4

אחדות מהוראות הפלט שלפניך שגויות. סמן אותן והסבר מהן הטעויות.

	משפט	לא שגוי	שגוי
1.	`alert("Programming in JavaScript");`		
2.	`alert(hello, world!);`		
3.	`alert("")`		
4.	`alert("alert(hello, world!);");`		
5.	`Alert "We have not yet begun";`		
6.	`alert("function main()");`		
7.	`alert(" we're leaving together ");`		
8.	`Alert('Alert! Alert!');`		

הסבר לשורות השגויות:

- בדרך כלל, בכתיבת קוד דרושות הערות. את ההערות ניתן להוסיף בשתי דרכים כפי שמודגם בדוגמה זו.

```
// This is how you create a single line remark
שני לוכסנים יוצרים הערה בת שורה //
/* This is how you create a multiple row remark-
לוכסן וכוכבית פותחים הערה מרובת שורות וכוכבית ולוכסן סוגרים אותה*/
```

הערה

הערת HTML נראית כך:

```
<!--  bla, bla, bla  -->
```

הערה בתסריט נראית כך:

```
// bla, bla, bla
```

שאלה 2.5

מה תציג התוכנית הבאה:

```
<script>
 // alert ("Hello World!") ;
</script>
```

- כאשר רוצים לציין קו אלכסוני במחרוזת, יש להשתמש בקו / באופן הבא:

```
my File = "images/lopez.jpg"
```

פרויקט סיינפלד

ראה את המשך הפרויקט בפרק 5.

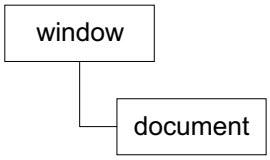
פרק 3

מתחילים לכתוב

הדרישות המוקדמות לפרק זה ולכל הפרקים שאחריו הוא הכרת שפת תכנות אחת. אם מושגים כמו משתנה, לולאה, אופרטור אינם מוכרים לך, אתה מוזמן לעיין בספר **JavaScript סדנת לימוד** בהוצאת **הוד-עמי**. גם אם אתה יודע שפת תכנות כלשהי ובמהלך הלימוד עם ספר זה תגיע למסקנה שאתה זקוק לרענון ו/או שדרוג, תוכל להיעזר בספר **JavaScript סדנת לימוד**.

אובייקט המסמך - document

זה הזמן להכיר את האובייקט הראשון שלנו - אובייקט המסמך, **document**. אובייקט זה מתאר את מסמך HTML החל מהתגית <body> ועד התגית </body>. הוא אינו האובייקט הראשון בהיררכיה. מעליו נמצא האובייקט window.



תרשים 3.1 : היררכיית האובייקטים.

לאובייקט document יש **אוספים** (collections), **שיטות** (methods), **תכונות** (properties) וגם אובייקטים הנמצאים מתחתיו.

השיטה bgColor

אחת התכונות של אובייקט document היא צבע הרקע של המסמך (להזכירך, צבע הרקע של המסמך נקבע בתגית <body> בעזרת המאפיין bgcolor). לתכונה של צבע הרקע ניגשים באופן הבא: document.bgColor. שים לב לאות C הכתובה כאות רישית!

ניתן להשתמש בתכונות בשתי דרכים:

- הצגת ערך התכונה:

```
<body bgcolor="blue">
<script language="javascript">
   window.alert(document.bgColor) ;
</script>
</body>
```

- קביעת ערך התכונה:

```
<script language="javascript">
   document.bgColor="red" ;
</script>
```

בדרך הראשונה, ביקשנו לקבל את ערך התכונה **bgColor** ולכן קבענו את ערך צבע הרקע בתגית <**body**> של HTML. לאחר מכן, השתמשנו בשיטה **alert()** להצגת התכונה. שים לב, שם התכונה נמצא בין הסוגריים ללא גרשיים!

נסה לתחום את התכונה בין גרשיים ("alert("document.bgColor, ותגלה כי שימוש בגרשיים הופך את תוכן ההוראה alert למחרוזת טקסט!

בדרך השנייה, קבענו את ערך התכונה (red) על ידי השימוש בסימן שווה, =.

הערה

bgColor היא תכונה של האובייקט document שהוא אובייקט של אובייקט window. לכן, ההיררכיה צריכה להירשם כך:

```
window.document.bgColor
```

אבל אפשר גם:

```
document.bgColor
```

באופן כללי ניתן שלא לציין את ההתייחסות לאובייקט window. אני מציע, כדי למנוע טעויות, לכתוב את כל ההיררכיה, אם זה אפשרי ולא מכביד. אני מעדיף את הדרך הארוכה, אך אני חייב להודות שלא תמיד אני נוהג כך:

```
window.document.bgColor
```

שימוש בשיטה ()write

האובייקט **document** מכיל שיטה ליצירת מסמך HTML ושמה הוא **()write**. בעזרתה נוכל לחולל מסמך בדרך שמוצגת בדוגמה הבאה.

הערה

()write היא שיטה של האובייקט document שהוא אובייקט של אובייקט window. לכן, ההיררכיה צריכה להירשם כך:

```
window.document.write()
```

אבל אפשר גם:

```
document.write()
```

באופן כללי ניתן שלא לציין את ההתייחסות לאובייקט window. אני מציע, כדי למנוע טעויות, לכתוב את כל ההיררכיה, אם זה אפשרי ולא מכביד. אני מעדיף את הדרך הארוכה, אך אני חייב להודות שלא תמיד אני נוהג כך:

```
window.document.write()
```

דוגמה 3-01.html:

```
<script language="javascript">
  document.write("<body bgcolor='red'
text='yellow'><h1>Hello</h1>") ;
</script>
```

ניתן אף להציג ערכי ביטויים או תכונות ללא שימוש בגרשיים. נעשה זאת כפי שמוצג בדוגמה הבאה.

דוגמה 3-02.html:

```
<body bgcolor="red">
   <script language="javascript">
       document.write(document.bgColor) ;
   </script>
</body>
```

נסה להריץ במחשב את הדוגמאות! התוצאה שהתקבלה, #ff0000, היא הייצוג ההקסדצימלי של הצבע red במבנה RGB.

◆ שאלה 3.1

מה יציג דף HTML הבא:

```
<html>
  <head>
   <title></title>
  </head>
  <body bgcolor="black">
    <script language="JavaScript">
      alert(document.bgColor);
    </script>
  </body>
</html>
```

סמן את התשובה הנכונה:

א. black

ב. #000000

ג. document.bgColor

ד. null

שרשור

כאשר רוצים לחבר שני איברים שונים או יותר ב-JavaScript, כמו למשל מחרוזת טקסט וערך תכונה, משתמשים בסימן השרשור +.

דוגמה 3-03.html:

```
<body bgcolor="red">
   <script language="javascript">
     document.write("The value of the background color is " +
     document.bgColor) ;
   </script>
</body>
```

בדוגמה זו שרשרנו את מחרוזת הטקסט "The value of the background color is " לתכונה **bgColor** של האובייקט **document**. נסה להריץ במחשב את הדוגמה.

הערה

הסימן ↳ נמצא בשימוש רק בספר ופירושו, ששורה זו הינה המשך ישיר ורציף של השורה הקודמת. את ההמשך יש להקליד ברצף וללא לחיצת Enter. לחיצת Enter רק לאחר התו ; (נקודה-פסיק).

משתנים

משתנים (variables) הם תאי זיכרון השומרים ערכים. לכל משתנה יש שם המאפשר גישה אליו. לדוגמה, ניתן ליצור משתנה בשם someNumber ולאחסן בתוכו את הערך 12. לאחר מכן, ניתן לבצע חישוב אריתמטי המתייחס למשתנה זה. לדוגמה:

```
someNumber + 5
```

התוצאה תהיה 17, מכיון שערכו של משתנה someNumber בתחילת החישוב היה 12.

שמות משתנים

בכתיבת שמות יש לנהוג לפי כללים מקובלים.

- התו הראשון חייב להיות אות (a..z, A..Z).
- שם יכול להיות מחרוזת תווים, החל באות אחת ועד עשרות אותיות. מותר להשתמש באותיות האלף בית הלועזיות (a..z, A..Z), בספרות (0..9) ובשני תווים מיוחדים: _ ו-$.
- אסור להשתמש **ברווח** (space character).
- אסור לעשות שימוש במילה שמורה.

טבלה 3.1: מילים שמורות ב-JavaScript

abstract	else	instanceof	switch
boolean	enum	int	synchronized
break	export	interface	this
byte	extends	long	throw
case	false	native	throws
catch	final	new	transient
char	finally	null	true
class	float	package	try
const	for	private	typeof
continue	function	protected	var
debugger	goto	public	void
default	if	return	volatile
delete	implements	short	while
do	import	static	with
double	in	super	

מומלץ לקבוע למשתנים שמות בעלי משמעות המתארים את מהות המשתנה ואת תפקידו. בנושא זה התפתחו מספר שיטות למתן שמות למשתנים, כאשר מטרתם אחת: להפוך את התוכנית שלך קלה יותר לכתיבה ולקריאה. אני לא נגד השיטה בה קוראים למשתנים בשמות כמו i, j או a, אלא שאני חושב שכדאי להשתמש בשם קצר שכזה רק במקרים שהמשתנה "חי" (במובן שהוא נמצא בזיכרון ואפשר להתייחס אליו) במסגרת המשפט בו הוא הוגדר ואולי שניים אחריו. למשל, בלולאת for המשתרעת על פני 4 או 5 שורות, כדאי להשתמש במשתנה-קצר-שם במקרה בו הוא משמש לאיחסון רגעי או זמני ולא כדאי להשקיע זמן במתן שם בעל משמעות, מכיון שזה אורך זמן ואין לכך חשיבות. למשתנה שנעשה בו שימוש לאורכו של הקוד יש לתת שם בעל משמעות.

השיטה ההונגרית (Hungarian convention) מוסיפה קידומת לפני שם המשתנה המציינת את סוג הנתון שבו. למשתנה שיכיל מחרוזת נוסיף את הקידומת str, למשל: strName, strHelpMessage, strSaleRegion. למשתנה שיכיל מספר שלם הקידומת תהיה int, למשל: intLoop, intZipCode, intSaleReqionQuantity. קידומות נוספות: bln (boolean, בוליאני), sng או flt (floating-point, נקודה-צפה).

בשיטה הנקראת formated compound words כל אות ראשונה היא אות ראשית, למשל: MaxiumWeight ו- FindHeightInMeters.

בווריאציה של השיטה שלעיל, המילה הראשונה תמיד תהיה באותיות קטנות ואחר כך כל אות ראשונה במילה היא אות ראשית, למשל: maxiumWeight ו- findHeightInMeters.

יש גם המפרידים בין מילה למילה בקו תחתי, למשל: Maxium_Weight ו- Find_Height_In_Meters.

תן למשתנים שמות בשפה האנגלית כמו myPicture ולא HatmunaSeli, מכיון שגם לך יהיה קשה לקרוא את הקוד שכתבת ולא רק לאחרים. שים לב שהשמות לא יהיו "ארוכים" מדי. למרות שאין מגבלה לאורך שם משתנה, נסה לצמצם ולתת שם שתוכל לעבוד איתו ולהבין את תפקידו בתוכנית.
במקום משתנה בשם strUserPerviosName רשום strUserPrevName, במקום intMaximumMessagesToBeSend רשום intMaxMsg2bSend, במקום fltTemporaryPrice רשום fltTmpPrice.

לא חשוב איזו שיטה תבחר, העיקר שתהיה עקבי.

שאלה 3.2

סמן נכון/לא נכון:

	נכון	לא נכון	
1.			השימוש במשתנים מאפשר קליטת ערכים מהמשתמש וביצוע חישובים שונים.
2.			כל משתנה תופס שטח ניכר מזיכרון המחשב.
3.			בשפת JavaScript יש סוג משתנה אחד ויחיד.
4.			כל משתנה מיוצג על ידי שמו, אותו ניתן להרכיב מאותיות לועזיות, מספרות ומקו תחתי.
5.			כל המשתנים בשפת JavaScript תופסים שטח זהה בזיכרון המחשב.
6.			אין הגבלה על השמות שניתן להעניק למשתנים.

הסבר את המשפטים הלא נכונים:

שאלה 3.3

סמן את שמות המשתנים הבלתי חוקיים בעיגול, והסבר מה לא נכון:

	חוקי/לא חוקי	למה לא חוקי?
1stMonth		
taxRate		
ThisIsMyLife_WOW		
lovely day		
loto$		
_familyName		

הגדרת משתנה

התחביר להגדרה/יצירת משתנה הוא:

```
var vItem ;
```

המילה השמורה var יוצרת את המשתנה, אחריה יש לכתוב את שם המשתנה ולבסוף נקודה-פסיק (זכור שלא חובה לשים נקודה-פסיק והדפדפן יסתדר גם בלעדי תו זה, אבל אם תשים אותו, תדע מתי המשפט נגמר). תוך כדי משפט ההגדרה, ניתן גם לבצע הצבת ערך. כדי להציב את המספר 12 בתוך משתנה vItem, יש לרשום:

```
var vItem = 12 ;
```

הגדרה והצבה בעזרת הסימן = (שווה). כאמור, אין צורך להציב בו ערך אם לא דרוש, מכיון שניתן להגדיר משתנה ללא ערך התחלתי כלשהו.

הערה

משתנה שיוגדר באמצעות var ולא יוצב בו ערך יקבל את הערך undefined.

שפת JavaScript היא מסוג השפות הנקראות **loosely type language**. המשמעות בנוגע להגדרת משתנים היא שלא חובה להגדיר משתנה. בפעם הראשונה שהדפדפן ייתקל בשם משתנה, הוא יגיד לעצמו "אהה, הנה משתנה חדש!" ויגדיר אותו עבורך. בהמשך התוכנית כשהוא ייתקל שוב באותו שם משתנה, הוא יאמר לעצמו: "אהה, אני כבר מכיר אותך". מטעמים של נוהל כתיבת קוד נכון, יש להקפיד על הגדרת המשתנים במשפט var.

אפשר לכתוב:

```
var majorVer ;
majorVer = 4 ;
```

ואפשר לכתוב:

```
var majorVer = 4 ;
```

דוגמה 3-04.html:

```
<html dir="rtl">
 <head>
   <title>3-04</title>
   <script language="javascript">
     var dogAge = 4 ;
     document.write("הכלב שלי הוא בן " + dogAge) ;
   </script>
  </head>
<body>
</body>
</html>
```

ערכי משתנים

משתנים ב-JavaScript יכולים לקבל מספר רב של סוגי ערכים. משתנים אלה נקראים **משתנים חופשיים** (free variables). לפניך פירוט סוגי המשתנים המותרים:

- **מחרוזת טקסט (string)**: מספר תווים בין גרשיים או בין גרש בודד.

```
strCustomerPrase = "This is my night" ;
productName = 'web guide magazine' ;
```

- **ערך מספרי**: מספר הוא כל ערך חיובי או שלילי עד 10e307! (10 בחזקת 307). לאחר מכן, JavaScript מתייחסת למספרים כ-Infinity (אינסוף).

```
nPointDown = -9  ;
interestYearRate = 10.032 ;
```

- **ערך בוליאני**: משתנה יכול לקבל ערך true (אמת, נכון, כן) או ערך false (שקר, לא נכון, לא).

```
blnGo = true  ;
tryAgain = false ;
```

- **ערך undefined**: אם רק מכריזים על משתנה באמצעות var ולא מציבים בו ערך, קיים בו הערך undefined. מבחינת מספר (number) הוא לא מספר, כמחרוזת ערכו יהיה undefined וכמשתנה בוליאני ערכו יהיה false.

לצורך תרגול, בצע את הדוגמה הבאה המשלבת סוגי ערכים שונים:

דוגמה 3-05.html:

```
<html dir="rtl">
 <head>
  <title>3-05</title>
  <script language="javascript">
    var personName = "ירון" ;
    var personAge = 27 ;
    var married = true ;
    document.write(personName + " בן " + personAge +
     ↳ "<br />") ;
    document.write( "לשאלה 'האם הוא נשוי?' התשובה היא " +
     ↳ married) ;
      </script>
  </head>
</html>
```

ניתן לוותר על המילה var, אך נוח יותר להבין קוד בו מציינים אותה כשמגדירים משתנה חדש.

ניתן להגדיר מספר משתנים באותה שורה על ידי שימוש בפסיקים, כגון:

```
var sayHello="hello", visitorNo=14, knownSurfer=false ;
```

עם זאת, מומלץ להגדיר כל משתנה בשורה נפרדת לשם הסדר ולמען הקריאות (readability), כך:

```
var sayHello ;
var visitorNo ;
var knownSurfer ;
sayHello = "hello" ;
visitorNo = 14 ;
knownSurfer = false ;
```

שאלה 3.4

מצא את הטעות בכל אחת משורות ההגדרה הבאות:

		במידה ונפלה טעות איך המשפט צריך להיכתב
1.	`power ;`	
2.	`num1 var;`	
3.	`var pow`	
4.	`var x y z;`	
5.	`var, i, j;`	
6.	`var a, b, var c;`	

פעולות עם משתנים

ניתן לבצע פעולות מתמטיות על משתנים המכילים מספרים. למשל:

```
checkNumber = 14 ;
checkNumber = checkNumber + 3 ;
```

המשפט הראשון מציב את הערך 14 במשתנה בשם checkNumber. המשפט השני הוא משפט הצבה. בצידו הימני נלקח הערך של משתנה checkNumber השווה 14 ונוסף לו הערך 3. התוצאה מוצבת במשתנה checkNumber "ודורסת" את הערך שהיה קיים בה קודם. כלומר, עכשיו הערך במשתנה checkNumber הוא 17.

```
checkNumber = "14" ;
checkNumber = checkNumber + 3 ;
```

האם אתה יכול לשער מה תהיה תוצאת שני המשפטים האלה? במשפט ההצבה הראשון מוצבת המחרוזת "14" במשתנה בשם checkNumber. זוהי המחרוזת "14" ולא המספר 14. במשפט ההצבה השני נעשה ניסיון לבצע פעולה על מחרוזת "14" ומספר 3. התוצאה תהיה לא "+" במובן חיבור אלא "+" במובן שירשור, והתוצאה תהיה "143".

זכור, לא ניתן לבצע פעולות חיבור וחיסור ולקבל תוצאה נכונה עם משתנה מסוג מחרוזת טקסט, אפילו אם המחרוזת מכילה מספר. יש צורך להפוך את המחרוזת למספר, טרם ביצוע חישובים מתמטיים (על כך בפרק 7).

שאלה 3.5

מה ערכו של המשתנה pan בסיום סדרת הפקודות הבאה:

```
<script
language="javascript">
 pan = 3 * 7 + 1 ;
 Miri = "5" ;
 pan = pan + Miri ;
</script>
```

```
<script
language="javascript">
 pan = 3 * 7 + 1 ;
 Miri = 5 ;
 pan = pan + Miri ;
</script>
```

הסבר את התוצאות השונות?

אופרטורים

אופרטורים הם סימנים המשמשים לבדיקה ולקביעת ערך. לפניך רשימות חלקיות של האופרטורים הקיימים ב-JavaScript.

אופרטורים מתמטיים:

אופרטור	תיאור
+	חיבור
-	חיסור
*	כפל
/	חילוק
%	מודולוס, המציין שארית של פעולת חילוק (לדוגמה: 23%7=2).

אופרטורים להשוואת ערכים:

אופרטור	תיאור
>	true אם ערכו של האיבר השמאלי גדול מערכו של האיבר הימני, false בכל מקרה אחר.
>=	true אם ערכו של האיבר השמאלי גדול מערכו של האיבר הימני או שווה לו, false בכל מקרה אחר.
==	true אם ערכו של האיבר השמאלי שווה לערכו של האיבר הימני, false בכל מקרה אחר.
!=	true אם ערכו של האיבר השמאלי אינו שווה לערכו של האיבר הימני, false בכל מקרה אחר.
<=	true אם ערכו של האיבר השמאלי קטן מערכו של האיבר הימני או שווה לו, false בכל מקרה אחר.
<	true אם ערכו של האיבר השמאלי קטן מערכו של האיבר הימני, false בכל מקרה אחר.

אזהרה!

אופרטור ההשוואה == וסימן השוויון = הם לא אותו דבר!

אופרטורים לוגיים:

אופרטור	תיאור
&&	וגם, AND (צירוף של תנאים: כל התנאים צריכים להתקיים).
||	או, OR (לפחות תנאי אחד צריך להתקיים).

ניתן לבדוק קיום של מספר תנאים בדרך זו:

```
(aNumber > 1 && aNumber < 10)
```

כלומר, אם aNumber גדול מ- 1 **וגם** aNumber קטן מ- 10.

בדיקה לקיומו של תנאי אחד לפחות תתבצע כך:

```
(codeItem < 1 || codeItem > 10)
```

כלומר, אם codeItem קטן מ- 1 **או** גדול מ- 10.

חומר עזר

רשימה מלאה של כל האופרטורים נמצאת בספר **JavaScript סדנת לימוד** בהוצאת **הוד-עמי**, ובספר **JavaScript המדריך השלם** (מעמוד 167 עד 210) בהוצאת **הוד-עמי**.

קביעת ערך

ניתן לקבוע ערך למשתנה במספר דרכים. לדוגמה, במקום לכתוב:

```
a = a + 1 ;
```

תוכל לכתוב:

```
a++ ;
```

JavaScript עושה שימוש גם באופרטורים מיוחדים הנקראים **shorthand operators**. שים לב, כל המשפטים **באותה** שורה עושים בדיוק את אותו הדבר:

```
a = a + 1 ;     a += 1 ;     ++a    a++
a = a – 1 ;     a -= 1 ;     --a    a--
a = a + b ;     a += b ;
a = a – b ;     a -= b ;
a = a * b ;     a *= b ;
a = a / b ;     a /= b ;
a = a % b ;     a %= b ;
```

הערה

שימוש ב- shorthand operators מקובל בשפות תכנות כמו ++C, C, Java.

שאלה 3.6

מצא משפטי השמה מתאימים למשפטי ההשמה הבאים, תוך שימוש באופרטורי השמה מקוצרים:

		תשובות
1.	`home = home + 10 ;`	
2.	`price = price * 1.205 ;`	
3.	`exp = exp + 1 ;`	
4.	`vol = vol * vol * vol ;`	
5.	`kg = kg * (3 + mass) ;`	
6.	`min = min - 1 ;`	
7.	`col = col % (7 * row) ;`	
8.	`month = month - tot + 10 ;`	

שאלה 3.7

אחדים ממשפטי ההשמה שלפניך שגויים. סמן אותם והסבר מהן הטעויות.

		מה שגוי
1.	`distance + = 100 ;`	
2.	`temp++ ;`	
3.	`(faran + 5)-- ;`	
4.	`a + b %= c ;`	
5.	`eq =* 3.33 ;`	
6.	`highest ++ ;`	
7.	`pass++++ ;`	
8.	`weight*=100 ;`	
9.	`--avg-- ;`	
10.	`phys++math ;`	

שאלה 3.8

לפניך מספר משפטי השמה. מצא מהו הערך המוצב במשתנה בכל אחד מהם.

להלן הערכים ההתחלתיים של המשתנים:

salary	100
base	4
max	5
num	7

		ערך
1.	`value = 10 – 4 ;`	
2.	`tax = 0.19 * salary ;`	
3.	`extra = 17 % 3 ;`	
4.	`sqr = base * base ;`	
5.	`min = -max + 1 ;`	
6.	`length = 3 % 10 * 2 ;`	
7.	`div = 9 / 4.5 / 2 ;`	
8.	`avg = 4 % 2 + - 7 ;`	
9.	`delta = 4 - -num * max ;`	
10.	`len = 5 / (4 % 3) - 7 + 5 ;`	
11.	`num = num * num ;`	
12.	`sum = sum – sum ;`	

פרויקט סיינפלד

ראה את המשך הפרויקט בפרק 5.

פרק 4

לולאות ובקרת זרימה

לולאה היא דרך לבצע פעולה או רצף פעולות מספר פעמים מוגדר, וכמו כל שפת תכנות גם ל-JavaScript יש מספר מבני לולאות. בלולאה תוכל להיעזר להצגת מחרוזת טקסט חמש פעמים (למשל), למעבר על איברי מערך ועוד וכל זאת ללא צורך לכתוב את הפקודה יותר מפעם אחת.

הערה

פרק זה מיועד למי שיודע שפת תכנות כלשהי (Pascal, C, ++C וכדומה). ההנחה היא שאתה יודע מהי לולאה ומהו משפט if. נכון, לא בשפת JavaScript, אבל בשפת תכנות אחרת. אם דרוש לך רקע תכנותי, תוכל למצוא אותו בספר **JavaScript סדנת לימוד** בהוצאת **הוד-עמי** כדי ללמוד את יסודות התכנות עם JavaScript.

הלולאה while

התחביר לכתיבת הלולאה **while** הוא:

```
while(condition)
 {
   JavaScript statement(s)
 }
```

כלומר, **כל עוד** (while) מתבצע **התנאי** (condition) שרשום בין הסוגריים, יתבצעו הוראות JavaScript שכתובות בין זוג הסוגריים המסולסלים {}. לאחר ביצוע ההוראות ייבדק שוב התנאי. אם התנאי ימשיך להתקיים, יבוצעו שוב ההוראות שבין הסוגריים המסולסלים ושוב ייבדק התנאי. כך תבוצענה ההוראות שוב ושוב, עד אשר יחדל התנאי להתקיים. במצב זה תמשיך התוכנית לאחר הסוגר המסולסל { שבסוף קבוצת ההוראות.

הבט בדוגמה הבאה:

```
<script language="javascript">
   var counter = 1 ;
   while (counter < 5)
     {
       document.write(counter + "<br />") ;
     }
</script>
```

בדוגמה זו, נוצר משתנה בשם counter שערכו המספרי 1. כל עוד ערכו של counter קטן מ- 5, ערך זה יוצג על המסך. הלולאה שבדוגמה זו תפעל לנצח (לולאה אין-סופית), כי בכל פעם שהתנאי ייבדק הוא יתקיים (ערכו true) וההוראות תבוצענה שוב.

אם אתה רוצה להגביל את מספר הריצות של הלולאה, יש לדאוג שהתנאי יחדל להתקיים. עשה זאת על ידי שינוי ערך משתנה counter.

דוגמה 4-01.html:

```
<html>
 <head>
  <title>4-01</title>
  <script language="javascript">
   var counter ;
   counter = 1 ;
   while (counter < 5)
   {
    document.write(counter + "<br />") ;
    counter = counter + 1 ;
   }
  </script>
 </head>
<body>
</body>
</html>
```

בדוגמה זו יש ביטוי המגדיל את ערך counter ב- 1 בתוך תחום הלולאה **while** שבין הסוגריים המסולסלים (בין } לבין {). לכן, הלולאה תפעל ארבע פעמים בלבד כי בכל ריצה יגדל ערך counter, וכשערך זה יגיע ל- 5 התנאי לא יתקיים - ערכו יהיה false.

חומר עזר

רשימה מלאה של כל האופרטורים נמצאת בספר **JavaScript סדנת לימוד** בהוצאת **הוד-עמי**, ובספר **JavaScript המדריך השלם** (מעמוד 167 עד 210) בהוצאת **הוד-עמי**.

הצבת ערך במשתנה

בדוגמה **4-01.html** השתמשנו בביטוי counter=counter+1; כדי להגדיל את ערכו של counter ב-1. בדרך זו ניתן לבצע כל פעולה אריתמטית, כגון:

```
vSum = 5 ;
vSum = vSum * 19 ;
salePrice = vSum / 2 ;
```

דוגמה 4-02.html:

```
<html>
 <head>
  <title>4-02</title>
  <script language="javascript">
   var a ;
   var aCounter ;
   a = 4 ;
   aCounter = 0 ;
   while (a <= 100)
   {
    document.write(a + ",") ;
    a = a + 4 ;
    aCounter = aCounter + 1 ;
   }
   document.write("<hr />There are " + aCounter + " numbers
    ↳ that can be divided by 4 between 1 to 100") ;
  </script>
 </head>
<body>
</body>
</html>
```

תוכנית זו מציגה את כל הכפולות של 4 בין 1 ל- 100, סופרת ומציגה אותן. לשם כך נעשה שימוש בשני משתנים: משתנה a מקבל את ערכי כפולות 4 על ידי הגדלת ערכו ב- 4 בכל ריצה של הלולאה, ומשתנה aCounter סופר את מספר הפעמים שהלולאה רצה על ידי הגדלת ערכו ב- 1 בכל ריצה של הלולאה.

תנאי הלולאה הוא, כל עוד ערכו של משתנה a קטן או שווה 100.

במהלך ריצת הלולאה מוצג ערכו של a המשורשר אל המחרוזת המכילה פסיק, כדי להפריד בין המספרים. כמו כן, עולה ערכו של a ב- 4 כדי להגיע לכפולה הבאה של 4, וכן עולה ערכו של aCounter ב- 1 כדי לספור את מספר הפעמים שהלולאה רצה. לאחר סיום ריצת הלולאה (כאשר התנאי חדל להתקיים), מודפס קו הפרדה (התגית <hr />) ולאחריו מחרוזת המשלבת את ערכו של aCounter, כדי להציג את מספר הפעמים שהלולאה רצה. הלולאה מציגה את מספר הכפולות של 4 בין 1 ל- 100.

אופרטור מקוצר

ניתן לקבוע ערך למשתנה במספר דרכים, ואחת מהן היא להשתמש באופרטור מקוצר.
בדוגמה **4-03.html** יש שימוש באופרטור מקוצר בהפעלת השיטה ()write:

```
<html>
 <head>
  <title>4-03</title>
  <script language="javascript">
   var loopIndex ;
   loopIndex = 1 ;
   while (loopIndex < 5)
   {
    document.write(loopIndex++ + " ") ;
   }
  </script>
 </head>
<body>
</body>
</html>
```

והתוצאה:

```
1 2 3 4
```

שים לב שכאשר מציגים משתנה, אפשר גם לשנות את ערכו, כמו שנעשה בתסריט שבדוגמה **4-03.html**. הפעולה loopIndex++ גרמה לכך שערכו של המשתנה loopIndex יישלח לשיטה ()write ואחר כך יתווסף לערכו הערך 1.

המשפטים הבאים זהים:

`document.write(loopIndex++) ;`	`document.write(loopIndex) ;` `loopIndex = loopIndex + 1 ;`

בתסריט שבדוגמה **4-04.html** מודגמת פעולת ההוספה לפני הצגת הערך של המשתנה. הפעולה loopIndex++ גרמה לכך שלערכו של המשתנה loopIndex יתווסף הערך 1 ואחר כך ערכו יישלח לשיטה ()write.

המשפטים הבאים זהים:

`document.write(++loopIndex) ;`	`loopIndex = loopIndex + 1 ;` `document.write(loopIndex) ;`

השווה בין התוצאות השונות של דוגמאות **4-03.html** ו- **4-04.html**.

דוגמה 4-04.html:

```
<html>
 <head>
  <title>4-04</title>
  <script language="javascript">
   var loopIndex ;
   loopIndex = 1 ;
   while (loopIndex < 5)
   {
    document.write(++loopIndex + " ") ;
   }
  </script>
 </head>
<body>
</body>
</html>
```

והתוצאה:

```
2  3  4  5
```

הערה

בקבצים **4-03.html** ו- **4-04.html** שורשרה המחרוזת   לערך המשתנה loopIndex וזאת כדי להציג תו רווח. בהצגת תו רווח בודד אפשר היה שלא לכתוב   ופשוט להשאיר רווח, באופן הבא:

```
document.write(++loopIndex + " ") ;
```

כדי להציג יותר מרווח אחד יהיה עליך לציין רווח קשיח.
להצגת שלושה תווי רווח, רשום:

```
document.write(++loopIndex + "   ") ;
```

פרטים נוספים על הצגת תווים מיוחדים, ראה בספר **מבוא לתכנות בסביבת אינטרנט, מבוא ו-HTML**, פרק 3.

הלולאה for

הנה התחביר של לולאת **for**:

```
for (initial variable value ; condition ; variable change)
   {
      JavaScript statement(s)
   }
```

בחלק הראשון של ההוראה for מוגדר משתנה הלולאה וניתן לו ערך התחלתי. חלק זה מתבצע פעם אחת עם תחילת הלולאה. החלק השני מכיל את התנאי להפסקת פעולתה של הלולאה. בתנאי נבדק ערכו של משתנה הלולאה ביחס לערך בכל מחזור ביצוע הלולאה. בחלק השלישי של הלולאה ישנו משפט השמה לקידום משתנה הלולאה שגם הוא מתבצע בסיום כל מחזור.

חלקי הלולאה מופרדים ביניהם בנקודה-פסיק (;).

בעת הכניסה ללולאה מוצב הערך ההתחלתי במשתנה הלולאה. תנאי הלולאה נבדק. אם הוא מתקיים, אז מבוצעים משפטי הלולאה. משתנה הלולאה מקודם, תנאי הלולאה נבדק וחוזר חלילה כל עוד ביטוי הלולאה מתקיים.

הערה

בדרך כלל, ובעיקר בגלל סיבות היסטוריות שלא כאן המקום לפרטן, קוראים למשתנה הלולאה בשם i. עד כדי כך השתרש השימוש במשתנה i בלולאה for, עד כי יש אנשים הקוראים ללולאה בשם for i (פור איי). אין לולאה for i, יש לולאת for.

דוגמה 4-05.html:

```
<script language="javascript">
   var i ;
   for(i = 1 ;  i < 5 ;  i++)
   {
    document.write(i + "<br />") ;
   }
</script>
```

שים לב שהפרמטרים של הוראת הלולאה for יכולים להיות ערכים קבועים, או משתנים ממש. אין חובה להגדיר את משתנה i לפני ביצוע הלולאה באמצעות המשפט var, מכיון ששפת JavaScript היא שפה חופשית (Loosely Type Language). מי שרוצה יכול לכתוב גם כך:

```
for(var i = 1 ;  i < 5 ;  i++)
```

הפרמטר האחרון בהוראה for מציין את שינוי ערך משתנה הלולאה בסיומו של כל מחזור ביצוע הלולאה. בשל הגדרת הפעולה בתחביר הלולאה, אין צורך לציין שינוי ערך כחלק מהוראות הלולאה, כפי שנעשה בלולאה מסוג while.

לולאה בתוך לולאה - לולאות מקוננות

לעיתים יש צורך לשלב לולאה בתוך לולאה. במצבים כאלה כדאי להקפיד על כתיבה מסודרת, בה הוראות הלולאה החדשה נכתבות בהזחה מתחילת השורה.

דוגמה 4-06.html:

```
<script language="javascript">
   var rowNo ;
   var colNo ;
   for (rowNo = 1 ;  rowNo < 10 ;  rowNo++)
   {
    for(colNo = 1 ;  colNo < 3 ;  colNo++)
    {
     document.write("[" + rowNo + "," + colNo + "]" +
     ↳ "   ") ;
    }
    document.write("<br />") ;
   }
</script>
```

תרגיל:

הצג את לוח הכפל 10 x 10, עשר שורות ובכל שורה עשרה מספרים.

פתרון:

כשניגשים לכתיבת קוד יש להציג תחילה את אלגוריתם התוכנית. כלומר, יש לתכנן את מהלך זרימת הפעולות בתוכנית.

1. דרושה לולאה אשר תרוץ 10 פעמים עבור כל איבר בשורה, ואחריה תעבור לשורה הבאה (<br />).
2. דרושה לולאה אשר תדאג כי הלולאה הקודמת תפעל 10 פעמים, כדי להציג 10 שורות.

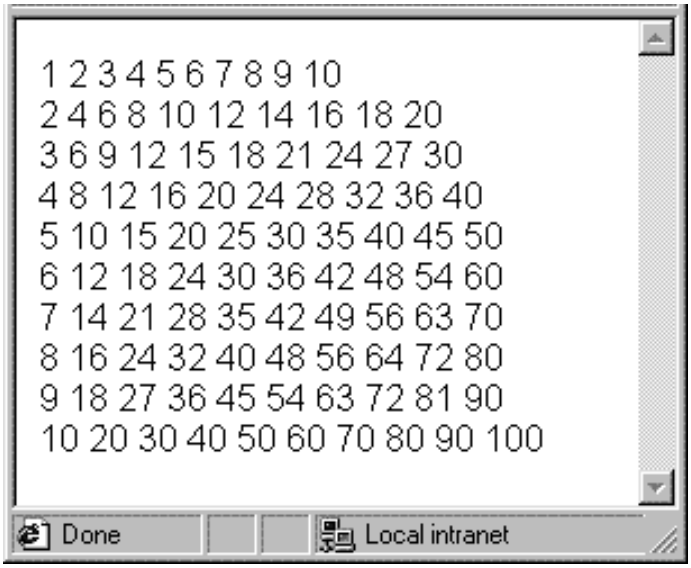
זוהי הלולאה המציגה שורה של 10 מספרים. משתנה הלולאה הוא משתנה i:

```
for (i =1 ;  i <= 10 ;  i++)
{
   document.write(??? + " ") ;  //   הכנסנו מחרוזת ריקה עבור רווח
}
document.write("<br />") ;       // לאחר הצגת 10 מספרים השורה תישבר
```

קטע קוד זה צריך לפעול 10 פעמים, ולכן הוא יירשם בתוך לולאה שתרוץ בעצמה 10 פעמים. במקום סימני השאלה נוכל לרשום את פעולת הכפל של משתנה הלולאה החיצונית (i), בערך משתנה הלולאה הפנימית (j).

דוגמה 4-07.html:

```
<script language="javascript">
   var i ;
   var j ;
   for(i = 1 ;  i <= 10 ;  i++)
   {
    for (j = 1 ;  j <=10 ;  j++)
    {
     document.write(i*j + " ") ;
    }
    document.write("<br />") ;
   }
</script>
```



תרשים 4.1: שתי לולאות מקוננות היוצרות את לוח הכפל (4-07.html).

חומר עזר

עוד על לולאות תוכל למצוא בספר **JavaScript סדנת לימוד**, בהוצאת **הוד-עמי**:

while	פרק 13
do..while	פרק 14
for	פרק 15

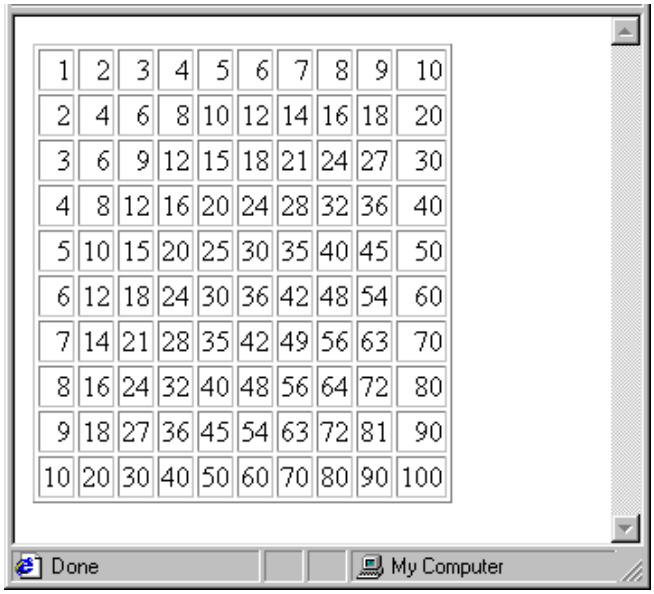
הכנסת מספרים לטבלת HTML

לוח הכפל שהתקבל בדוגמה **4-07.html** לא נראה טוב, אם כי תוכנו נכון. הצבתו של לוח הכפל בטבלה עשוייה לשפר את מראהו. התגיות של פתיחת הטבלה **<table>** יופיעו פעם אחת לפני תחילת הלולאות ופעם נוספת - כסגירה, לאחר סגירת הלולאה הגדולה, **</table>**. תגיות הפתיחה והסגירה של השורות, **<tr>** ו-**</tr>**, יופיעו במסגרת הלולאה הגדולה ויחליפו למעשה את שבירת השורה על ידי **
. גם תגיות הפתיחה והסגירה של תא בטבלה **<td> ו-**</td>** יופיעו במסגרת הלולאות.

דוגמה 4-08.html:

```
<script language="javascript">
   var i ;
   var j ;
   document.write("<table border='1' cellpadding='3'>") ;
   for (i = 1 ;  i <= 10 ;  i++)
   {
    document.write("<tr>") ;
    for (j = 1 ;  j <= 10 ;  j++)
    {
     document.write("<td align='right'>" + i*j +"</td>") ;
    }
    document.write("</tr>") ;
   }
   document.write("</table>") ;
</script>
```

וזו התוצאה:



1	2	3	4	5	6	7	8	9	10
2	4	6	8	10	12	14	16	18	20
3	6	9	12	15	18	21	24	27	30
4	8	12	16	20	24	28	32	36	40
5	10	15	20	25	30	35	40	45	50
6	12	18	24	30	36	42	48	54	60
7	14	21	28	35	42	49	56	63	70
8	16	24	32	40	48	56	64	72	80
9	18	27	36	45	54	63	72	81	90
10	20	30	40	50	60	70	80	90	100

תרשים 4.2: תוכנית JavaScript יוצרת קוד HTML (4-08.html).

התנאי if...else

כמו בכל שפת תכנות, ניתן לבצע בדיקות נקודתיות על ידי הפקודה **if**. כלומר, ניתן לבדוק קיום של תנא/ים ולהתנות ביצוע הוראה/ות מסוימות בתוצאות הבדיקה. להלן התחביר:

```
if (condition)
   {
    javascript statement(s)
   }
```

ביטוי התנאי יכול להיות true או false. אם התנאי, שכתוב בין הסוגריים, מתקיים, יתבצעו ההוראות שבין הסוגריים המסולסלים. אם אין סוגריים מסולסלים, תתבצע רק ההוראה הראשונה שאחרי התנאי!

ניתן להוסיף גם את הפקודה **else**, אשר קובעת אלו הוראות יתבצעו כאשר התנאי אינו מתקיים. בדוגמה הבאה ניתן לראות שהתנאי הוא האם ערך המשתנה a שונה (לא שווה) ל- 5. תנאי יכול לתת תשובה true או false. במקרה שהתשובה היא true, כלומר, ערכו של המשתנה a שונה מ- 5, יתבצע המשפט document.write() שיציג את ערכו של המשתנה a. כאשר התנאי אינו מתקיים, כלומר התשובה היא false, אז יתבצעו המשפט/ים מעבר למילה else. נקבל בביטוי התנאי תשובה false כאשר משתנה a לא

שונה מ-5, כלומר, שמשתנה a כן שווה 5, אז תודפס המילה Five. כל הפקודות האלו נמצאות בתחום הלולאה for.

דוגמה 4-09.html:

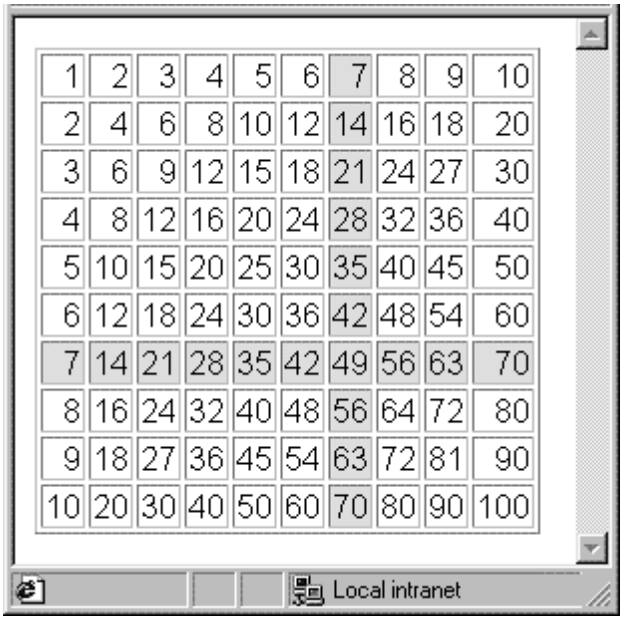
```
<script language="javascript">
   var i ;
   for (i = 1 ;  i <= 10 ;  i++)
   {
    if(i != 5)
    {
     document.write(i + "<br />") ;
    }else{
     document.write("Five<br />") ;
    }
   }
</script>
```

כעת נשתמש בבדיקות if ו-else כדי לשכלל את טבלת המספרים ולהוסיף צבע אדום בכל מספר שהוא כפולה של 7. כדי לוודא שמספר הוא כפולה של 7, עליו להתחלק ב- 7 ללא שארית, כלומר if (i%7==0), שהרי "i%7" נותן את השארית של חלוקת i ב- 7.

נכניס את התנאי לקוד הקודם, דוגמה **4-08.html**. אולם הפעם, במקום להדפיס את תא הטבלה בכל ריצה של הלולאה הפנימית, נבדוק ונחליט אם להציג את צבע רקע של התא בצבע חאקי (khaki) או בצבע רגיל (לבן). ראה דוגמה **4-10.html**.

דוגמה 4-10.html:

```
<script language="javascript">
   var i ;
   var j ;
   document.write("<table border='1' cellpadding='3'>") ;
   for (i = 1 ;  i <= 10 ;  i++)
   {
    document.write("<tr>") ;
    for (j = 1 ;  j <= 10 ;  j++)
    {
     if (i*j % 7 == 0)
     {
     document.write("<td bgcolor='khaki' align='right'>" + i*j +"</td>") ;
     }else{
      document.write("<td align='right'>" + i*j +"</td>") ;
     }
    }
    document.write("</tr>") ;
   }
   document.write("</table>") ;
</script>
```



1	2	3	4	5	6	7	8	9	10
2	4	6	8	10	12	14	16	18	20
3	6	9	12	15	18	21	24	27	30
4	8	12	16	20	24	28	32	36	40
5	10	15	20	25	30	35	40	45	50
6	12	18	24	30	36	42	48	54	60
7	14	21	28	35	42	49	56	63	70
8	16	24	32	40	48	56	64	72	80
9	18	27	36	45	54	63	72	81	90
10	20	30	40	50	60	70	80	90	100

תרשים 4.3: שילוב משפט if בלולאה for (4-10.html).

שים לב לדרך כתיבת התנאי, בה נעשה חישוב (i*j), בוצעה פעולה (%) ונעשתה השוואה (==):

```
if (i*j % 7 == 0)
```

אם זה יותר נוח לך תוכל להוציא את הפעולה מחוץ לתנאי, למשל:

```
tmp = i*j ;
if (tmp % 7 == 0)
```

או כך:

```
tmp = i*j % 7 ;
if (tmp == 0)
```

אתה יכול לבחור את דרך העבודה המתאימה לך.

תרגיל:

במקום לבדוק את שארית החלוקה ב-7 בדוק את שארית החלוקה ב- 4 למשל, וראה מה קיבלת.

חומר עזר

עוד על משפט if תוכל למצוא בספר **JavaScript סדנת לימוד**, פרקים 8, 9 ו- 10, בהוצאת **הוד-עמי**.

שאלה 4.1

לפניך מספר קטעי תוכניות. ציין מהו הפלט (אם יש) המתקבל מכל אחד מהם, עבור כל אחד מהערכים הרשומים אחריו (שים לב כי מספר טעויות נעשו בהזחת המשפטים).

```
if (x > 4)
  if (x > 7)
    alert("USA");
else
  alert("ISRAEL");
```

הפלט עבור x=10: _____

הפלט עבור x=0: _____

הפלט עבור x=5: _____

```
if (x % 3 == 0)
  if (x / 4 != 6)
    if (x < 20)
      alert("xyz");
     else
      alert("abcdefg");
```

הפלט עבור x=0: _____

הפלט עבור x=24: _____

הפלט עבור x=36: _____

```
if (x < 0) {
  if (x > -8)
    alert("Jason Alexander");
}
else
  alert("Jerry Seinfeld");
```

הפלט עבור x=-7: _____

הפלט עבור x=3: _____

הפלט עבור x=-9: _____

```
if (x > 100)
  alert("Sue Eillen Mischke");
else if (x > 10)
  alert("Crazy Joe Davola");
else if (x > 0)
  alert("Mr.Pitt");
else if (x < 0)
  alert("J.Peterman");
else
  alert("Uncle Leo");
```

הפלט עבור x=140: _____

הפלט עבור x=3: _____

הפלט עבור x=0: _____

הפלט עבור x=-4: _____

הפלט עבור x=13: _____

הפלט עבור x=100: _____

```
if (x % 2 == 0)
  if (x % 3 == 0)
    if (x % 5 == 0)
      if (x % 7 == 0)
        alert("Frank Constanza");
       else
        alert("Estelle Constanza");
     else
      alert("Jeorge Constanza");
```

הפלט עבור x=0: _____

הפלט עבור x=90: _____

הפלט עבור x=210: _____

הפלט עבור x=1: _____

הפלט עבור x=12: _____

הפלט עבור x=4: _____

שאלה 4.2

כתוב תוכנית המציגה על המסך טבלה בגודל 6x6.

א. גובה כל תא 30 נקודות ורוחבו 30 נקודות.

ב. צבע התאים בעמודות (משמאל לימין) הוא: red, blue, yellow, green, gray, pink.

ג. בשורה הראשונה רק תא אחד צבוע (התא השמאלי ביותר), בשורה השנייה רק שני תאים צבועים (השמאליים ביותר) וכך הלאה, עד שבשורה השישית כל התאים צבועים.

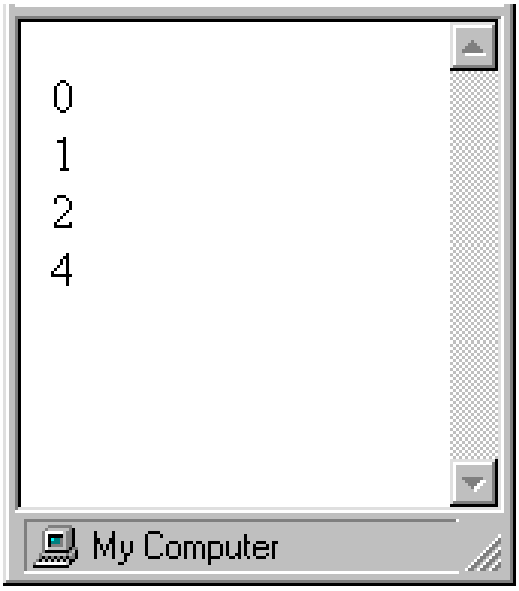
הפקודה continue

הפקודה **continue**, המופיעה במהלך ביצוע הפעלת הלולאה, ממשיכה את ביצוע הלולאה שוב מראש הלולאה. כלומר, היא מחזירה את סמן ריצת התוכנית אל תחילת הלולאה. בדוגמה הבאה תפעל הלולאה 5 פעמים ובכל פעם תדפיס את מספר ריצת הלולאה על ידי הדפסת משתנה הלולאה i. לפני הצגת ערכו של המשתנה i תתבצע בדיקת ערכו. אם ערכו של i שווה 3, תתבצע הפקודה continue. במקרה זה התוכנית לא תמשיך בביצוע הלולאה, אלא תדלג לראש הלולאה להמשך ביצוע. המספר 3 לא יודפס.

דוגמה 4-11.html:

```
<script language="javascript">
   var i ;
   for(i = 0 ;  i < 5 ;  i++)
   {
    if (i == 3) {
     continue ;
    }else{
     document.write(i + "<br />") ;
    }
   }
</script>
```

התוצאה: מספר 3 לא יודפס.



תרשים 4.4 : דוגמה לשימוש במשפט continue (4-11.html).

הפקודה break

הפקודה **break** מפסיקה את ריצת לולאה. אם נשתמש בדוגמה הקודמת ונחליף את הפקודה continue בפקודה break, נכפה את הפסקת פעולת הלולאה כאשר ערכו של i יגיע ל- 3.

דוגמה 4-12.html :

```
<script language="javascript">
   var i ;
   for(i = 0 ;  i < 5 ;  i++)
   {
    if(i == 3) {
     break ;
    }else{
     document.write(i + "<br />") ;
    }
   }
</script>
```

חומר עזר

עוד על משפטים break ו-continue תוכל למצוא בספר **JavaScript סדנת לימוד**, פרק 17, בהוצאת **הוד-עמי**.

בדיקת ערך בוליאני

תנאי של לולאת while ותנאי של משפט if מחזיר true או false. כאשר התנאי הנבדק הוא משתנה בוליאני, ניתן לוותר על ההשוואה לערכים אלה. נניח וקיים משתנה בשם boolFlag שהוצב בו הערך false. כדי לבדוק את ערכו של משתנה boolFlag במשפט if אפשר לכתוב:

```
if (boolFlag == true)
```

וניתן גם לכתוב:

```
if (boolFlag)
```

הבט בלולאה while שבדוגמה הבאה.

דוגמה 4-13.html:

```
<script language="javascript">
   var a ;
   var TryAgain ;
   a = 1 ;
   TryAgain = true ;
   while (TryAgain)
   {
    document.write(a + "<br />") ;
    a = a + 1 ;
    if (a < 5) {
     TryAgain = true ;
    }else{
     TryAgain = false ;
    }
   }
</script>
```

לפני תחילת הלולאה במשתנה TryAgain מוצב הערך true. זהו משתנה בוליאני (boolean variable).

ביטוי התנאי בלולאה הוא while (TryAgain) שהיה יכול גם להיכתב while(TryAgain == true), לפי בחירתך.

במהלך הלולאה מקודם ערכו של משתנה a ב- 1.

משפט ה-if בודק אם ערכו של a קטן מ- 5. אם ערכו של הביטוי הוא true, כלומר a אכן קטן מ- 5, אז מוצב הערך true במשתנה TryAgain כדי שהלולאה תמשיך בפעולתה. אם התשובה היא false, כלומר ערכו של a לא קטן מ- 5 (כלומר הוא גדול או שווה ל- 5), אז מוצב הערך false במשתנה TryAgain ומכיון שכך ייפסק ביצוע הלולאה.

משפט switch

שיטה חילופית לבדיקה היא שימוש בפקודה **switch**. ניתן להטיל על הפקודה switch לבדוק ערך ביטוי ולפעול בהתאם לערך שיימצא. זהו סוג מסויים של משפט if. בדוגמה הבאה נריץ לולאה מ- 0 עד 100 ונבדוק את ערך המשתנה i בדרך זו:

```
switch (i)  {
    case 25 :                    כאשר הערך במשתנה i שווה 25
break ;
   case 50 :                     כאשר הערך במשתנה i שווה 50
break ;
   default:                      כאשר אף משפט case לא התבצע
 }
```

לאחר כל משפט case נרשום את ההוראה/ות שיש לבצע ואת ההוראה break, כדי למנוע את מילוי ההוראות במקרי case העוקבים. ניתן לרשום מספר הוראות בשורות שבין משפט case אחד למשנהו, או עד סגירת הסוגריים המסולסלים התוחמים את ההוראה switch.

דוגמה 4-14.html:

```
<script language="javascript">
   var i ;
   for(i = 1 ;  i < 100 ;  i++)
   {
    switch(i) {
     case 25 :
       document.write("עשרים וחמש<br />") ;
       break ;
     case 50 :
       document.write(i + "<br />") ;
    }
   }
</script>
```

חומר עזר

עוד על משפט switch תוכל למצוא בספר **JavaScript סדנת לימוד**, פרק 11, בהוצאת **הוד-עמי**.

שאלה 4.3

אחדים ממשפטי ה-switch שלפניך שגויים. סמן אותם והסבר בקצרה מהי הטעות/יות.

```
num1 = 7 ;
num2 = 8 ;
value = 7 ;
switch (value) {
case num1:
       alert("value equals num1");
       break;
case num2:
       alert("value equals num2");
       break;
}
```

❑ נכון ❑ לא נכון

```
switch (13) {
case 11:
       alert("Eleven");
       break;
case 12:
       alert("Twelve");
       break;
case 13:
       alert("Thirteen");
       break;
}
```

❑ נכון ❑ לא נכון

```
age = 13 ;
switch (age) {
case 13:
       alert("You're thirteen years old. ");
       break;
case 16:
       alert("You're sixteen years old. ");
       break;
case 13:
       alert("Have you had your bar-mitsva yet? ");
       break;
}
```

❑ נכון ❑ לא נכון

```
n = 14 ;
switch (n % 2) {
case 0:
       alert("is even " + n);
       break;
case 1:
       alert("is odd " + n);
       break;
}
```

❑ נכון ❑ לא נכון

```
x = 'f' ;
switch x {
case 'd':
       alert("integer");
       break;
case 'f':
       alert("floating point");
       break;
case '%':
       alert("percentage");
       break;
}
```

❑ נכון ❑ לא נכון

```
month = 3 ;
switch (month) {
case 4:
       alert("April");
       break;
}
```

❑ נכון ❑ לא נכון

```
switch (day)
case 1:
       alert("Sunday");
       break;
case 2:
       alert("Monday");
       break;
case 3:
       alert("Tuesday");
       break;
case 4:
       alert("Wednesday");
       break;
```

❑ נכון ❑ לא נכון

שאלה 4.4

לפניך מספר משפטי if ו-switch. במידת האפשר, הפוך את משפטי ה-if למשפטי switch.
אם לא ניתן לבצע זאת, הסבר מדוע.

```
if (a == 2)
        alert("a == 2");
else if (a == 3)
        alert("a == 3");
else if (a == 4)
        alert("a == 4");
```

```
if (num > 0)
    alert("num is positive");
else if (num < 0)
    alert("num is negative");
```

```
if (a == b)
        alert("a equals b");
else if (a == c)
        alert("a equals c");
```

```
if (num == 0) {
  max = 25;
  min = 10;
  alert("num equals zero");
}
else if (another_num == 1) {
  max = 10;
  min = 7;
  alert("another_num equals one");
}
```

```
if (fact == 720)
        n = 6;
else if (fact == 120)
        n = 5;
else if (fact == 24)
        n = 4;
else if (fact == 6)
        n = 3;
```

```
if (q == 7.5)
        alert("7.5");
else if (q == 6.2)
        alert("6.2");
```

פרויקט סיינפלד

ראה את המשך הפרויקט בפרק 5.

פרק 5

פונקציות

כדי להבין את המשמעות האמיתית של הפונקציות ב-JavaScript נתרגל תחילה **פקודות משובצות/מוטבעות** (Embedded Script). הן משובצות כחלק מתגית HTML.

כבר הכרנו את האובייקט document ואת התכונה bgColor שלו, המייצגת את צבע הרקע של הדף. כעת, נאפשר למשתמש לשנות את צבע הרקע על ידי פקודה מוטבעת.

אירועים

דפדפן Internet Explorer מזהה שורה ארוכה של אירועים (events): פתיחת חלון, לחיצה בעכבר ועוד. גם לחיצה על **קישור** (link) היא אירוע הגורם לכך שהדפדפן יבצע את מה שרשום במאפיין href של התגית <a>:

```
<a href="http://www.hod-ami.co.il">
 הוד-עמי, נותנים לך ידע להצלחה
</a>
```

אירוע (event) נוסף הוא מעבר עכבר מעל אובייקט בדף. כאשר זה קורה, נטען event-handler בשם **onmouseover**. אם אתה רוצה שהדפדפן יתייחס לאירוע מסויים, למשל, במסגרת התגית <a>, אתה חייב לציין את אותו event-handler של האירוע ואת הפעולה שתתבצע בעקבותיו. בדפדפן Internet Explorer ניתן להחיל event-handler בשם onmouseover גם על אובייקטים נוספים, כגון <p>, ואחרים.

הערה

לכל תגית HTML יש את ה- events-handlers שלה. פירוט ה- events-handlers שניתן לשייך לכל אובייקט מפורטים בספר **JavaScript המדריך השלם**, בהוצאת **הוד-עמי**.

```
<a href="http://www.hod-ami.co.il"
onmouseover="document.bgColor='red';">
הוד-עמי, נותנים לך ידע להצלחה
</a>
```

כלומר, ברגע שמעבירים את סמן העכבר על טקסט הקישור (הוד-עמי, נותנים לך ידע להצלחה) תתבצע שורת הקוד הקובעת את ערך התכונה bgColor.

ומה יקרה כאשר תלחץ על הקישור? נכון, הדפדפן יטען את האתר שבכתובת http://www.hod-ami.co.il.

שים לב, המילה red היא מחרוזת טקסט ולכן צריך להקיפה בגרשיים. עם זאת, שימוש בגרשיים יפגע בגרשיים המקיפים את המשפט כולו, ולכן יש להשתמש בגרש בודד. ניתן לראות זאת בדוגמה הבאה.

כל פקודה מסתיימת בנקודה-פסיק (;) שלא חובה לשים אותה, אלא במקרה בו תרצה להפעיל יותר מפקודה אחת.

רשימת אירועים

Event Handler	Event
onabort	המשתמש עוצר טעינת תמונה.
onblur	האובייקט מאבד את המיקוד כאשר אובייקט אחר נבחר במקומו.
onchange	אובייקט שמאבד מיקוד נבדק, כדי לדעת אם חל שינוי בערכו ההתחלתי, ואז מופעל ההדק.
onclick	מתבצעת לחיצה על אובייקט לחיץ.
ondblclick	לחיצה כפולה, שנעשית בדפדפן מגירסה 4.
onerror	שגיאה בהרצת הקוד.
onfocus	פעולת הדק הפוכה מ-onblur. מופעל כאשר אובייקט נבחר ומקבל מיקוד.
onload	לאחר טעינת המסמך. נמצא בתחום תגיות body.
onmouseout	יציאת עכבר מתחום אובייקט.
onmouseover	מעבר עכבר על אובייקט. כדי להבטיח תמיכה בכל הדפדפנים כדאי לשתול את ההדק בתוך פקודת קישורית, למרות שבדפדפני Internet Explorer ניתן להשתמש בהדק זה גם על האובייקטים עצמם.
onmouseup	שחרור לחצן העכבר (לאחר שנלחץ מטה).
onmousedown	לחיצת לחצן העכבר.
onunload	יציאה מדף.

דוגמה 5-01.html:

```
<html dir="rtl">
 <head>
  <title>5-01</title>
 </head>
<body bgcolor="green" text="yellow">
 <h1>הוד-עמי, נותנים לך ידע להצלחה</h1>
 <hr />
 <br />
  העבר את הסמן
 <a href="http://www.hod-ami.co.il"
  onmouseover="document.bgColor='red';">
  כאן
 </a>
  כדי לשנות את צבע הרקע
</body>
</html>
```

event-handler נוסף הוא **onmouseout**. הדפדפן מזהה אירוע (event) של יציאת העכבר מתחום האובייקט. ניתן לצרף את ה- event-handler בשם onmouseout אל התסריט שבדוגמה **5-01.html**:

```
<html dir="rtl">
 <head>
  <title>5-01</title>
 </head>
<body bgcolor="green" text="yellow">
 <h1>הוד-עמי, נותנים לך ידע להצלחה</h1>
 <hr />
 <br />
 העבר את הסמן
 <a href="http://www.hod-ami.co.il"
  onmouseover="document.bgColor='red';"
  onmouseout ="document.bgColor='lightblue';">
  כאן
 </a>
 כדי לשנות את צבע הרקע
</body>
</html>
```

שאלה ותשובה!

מה ההבדל בין אירוע (event) לבין event-handler?

אירוע קודם לטעינת event-handler:

1. אירוע (event) של לחיצה בעכבר (mouse) טוען event-handler בשם onclick.
2. אירוע (event) של טעינת מסמך (load) טוען event-handler בשם onload.

שאלה ותשובה!

האם ניתן "לתפוס" את האירוע רק באובייקט בו הוא הופעל?

לא רק. ניתן "לתפוס" את האירוע גם בנקודה גבוהה יותר בהיררכיית האובייקטים במסמך, כולל האובייקט הגבוה ביותר הנוצר על ידי התגית <body>. במילים אחרות, ניתן להוסיף את המאפיין onmouseover בתגית <body> כדי "ללכוד" אירוע שהופעל באובייקט אחר במסמך. נושא זה אינו כלול בתוכנית הלימודים. מי שמעוניין מוזמן להציץ בספר **JavaScript+CSS+DOM למפתחי אתרים באינטרנט** בהוצאת **הוד-עמי**.

פונקציות

פונקציה היא הוראה, או מספר הוראות, בעלת תחום פעולה מוגבל בדרך כלל - ביצוע פעולה מוגדרת - והיא אינה נמצאת ברצף הפקודות של התוכנית. כבר הכרת פונקציות אחדות בשפת JavaScript שנקראות **שיטות** (methods). **שיטה** היא למעשה **פונקציה** בנויה מראש שהיא חלק מה-DOM וכדי להיות יותר ספציפי היא חלק מאחד האובייקטים של ה-DOM כמו window, document וכדומה. למשל, השיטה ()write של אובייקט document מציגה את הערך שנשלח אליה בדף המוצג בדפדפן. היא לא מעלה בריבוע, מוציאה שורש או משנה את סדר התווים - היא מציגה ערך בדף המוצג בחלון הדפדפן.

בהמשך תלמד לבנות פונקציה משלך, שלמרבה הפליאה תהפוך להיות חלק מה-DOM לפרק הזמן בו הדף בו רשומה הפונקציה "חי" במערכת (בפרק 9 תראה כמה זה נוח שהפונקציה שלך היא חלק מה-DOM).

כדי לגרום לביצוע ההוראות הכלולות במסגרת הפונקציה, צריך להפעיל אותה על ידי פנייה מפורשת אליה, פעולת "קריאה" (call).

תחביר הפונקציה:

```
function colorChange()
```

ההצהרה שהקוד שייך לפונקציה נעשית על ידי המילה השמורה function ואחריה שם הפונקציה וסוגריים ריקים. בסוגריים תשתמש, בשלב מאוחר יותר, להעברת נתונים (פרמטרים) אל הפונקציה.

את ההוראות הנכללות בפונקציה יש לתחום בין סוגריים מסולסלים:

```
function  colorChange()
{
 document.bgColor = 'red'  ;
}
```

מיקום הפונקציות בתוכנית

את הפונקציות אפשר לכתוב בכל מקום בין פקודות התוכנית, ובלבד שהן תיכתבנה בין תגית **<script>** לתגית **</script>**. עם זאת, כדי למנוע מצב בו תתבצע קריאה לפונקציה שעוד לא נטענה אל הדפדפן, רצוי מאוד לכתוב את הפונקציות במסגרת תגיות <head>.

כתוב את הקוד הבא. אל חשש! דוגמה זו לא תתן תוצאה!

דוגמה 5-02.html:

```
<html dir="rtl">
 <head>
  <script language="javascript">
   function bgcolor2red()
   {
    document.bgColor='red' ;
   }
  </script>
 </head>
<body>
 <div align="center">
  <font size="5">הוד-עמי, נותנים לך יד<font
    ↳ color="blue">ע</font> להצלחה
  </font>
  <hr />
 </div>
</body>
</html>
```

כפי שבוודאי שמת לב, בדוגמה זו לא התבצעה הוראת שינוי לצבע הרקע. כדי להפעיל את הפונקציה bgcolor2red(), יש לקרוא לה ולהפעיל אותה במפורש.

קריאה לפונקציה

ניתן להשתמש במספר event-handlers כדי לקרוא לפונקציה. בדוגמה זו נשתמש בתגית הקישור <a> ובמאפיין href. בקריאה לפונקציה צריך לציין את שם הפונקציה (והסוגריים הריקים שאחריה), אולם אם הקריאה לפונקציה נעשית דרך המאפיין href של התגית <a>, יש להוסיף את המילה **javascript:** לפני השם ובצמוד לו.

הוסף לדוגמה **5-03.html** את הקישור הבא:

```
לחץ
 <a href="javascript:bgcolor2red();">
     כאן
 </a>
     כדי לשנות את צבע הרקע
```

התוכנית המעודכנת נראית כמו זו שבדוגמה הבאה, קובץ **5-03.html**:

```
<html dir="rtl">
 <head>
  <title>5-03</title>
  <script language="javascript">
   function bgcolor2red()
   {
    document.bgColor='red' ;
   }
  </script>
 </head>
<body bgcolor="green" text="yellow">
 <h1>הוד-עמי, נותנים לך ידע להצלחה</h1>
 <hr />
 <br />
 לחץ
 <a href="javascript:bgcolor2red();">
  כאן
 </a>
 כדי לשנות את צבע הרקע
</body>
</html>
```

יש מקרה מיוחד בו אנו מפעילים פונקציה בשם **void(0)**, באופן הבא:

```
העבר את הסמן
<a href="javascript:void(0);"
     onmouseover="sayHello();"
     onclick="paintInBlue();">
 כאן </a>
כדי לומר שלום, או לחץ כדי לצבוע בכחול.
```

בלי להיכנס להסבר של ()void, המילה **כאן** שבמשפט שלעיל היא קישור. לפחות ויזואלית, היא נראית כמו כל קישור "כשר". אלא שלחיצה על הקישור לא תפנה ל-URL, שלא קיים, אלא תפעיל את האירוע onclick, אם קיים. אם האירוע onclick לא ייכתב במסגרת התגית <a> בה המאפיין href מוגדר **javascript:void(0)**, לחיצה על הקישור לא תעשה דבר (קובץ **void.html**):

```
<html dir="rtl">
 <head>
  <title>void</title>
  <script language="javascript">
   function bgcolor2red()
   {
    document.bgColor='red' ;
   }
  </script>
 </head>
<body bgcolor="green" text="yellow">
 <h1>הוד-עמי, נותנים לך ידע להצלחה</h1>
 <hr />
 <br />
 העבר את הסמן
 <a href="javascript:void(0);"
  onmouseover="bgcolor2red();"
  onmouseout="document.bgColor='orange';">
  כאן
 </a>
 כדי לשנות את צבע הרקע
</body>
</html>
```

בעת טעינת קובץ **void.html**, הרקע יהיה בצבע ירוק:

- העבר את הסמן מעל המילה **כאן**, לקבלת צבע רקע אדום.
- הזז את סמן העכבר מעל המילה **כאן**, לקבלת צבע רקע כתום.
- לחץ על הקישור (המילה **כאן**) ושום דבר לא קורה.

דוגמאות לקריאה לפונקציה באמצעות אירועים:

```
<img src="joe.gif" onabort="functionName();" />

<input type="text" onblur="functionName();" />

<input type="text" onfocus="functionName();" />

<input type="checkbox" onchange="functionName();" />

<input type="button" onclick="functionName();" />

<a href="javascript:void(0);" onmouseover="functionName();">
 <img src="loo.gif" />
</a>

<img src="loo.gif" onmouseover="functionName();" />

<a href="javascript:void(0);" onmouseout="functionName();">
 <img src="loo.gif" />
</a>

<img src="loo.gif" onmouseout="functionName();" />

<a href="javascript:void(0);" ondblclick="functionName();">
 hello
</a>

<input type="button" onmouseup="functionName();" />

window.onerror=functionName();

<body onload="functionName();">

<body onunload="functionName();">
```

שאלה 5.1

תרגיל זה מבוסס על תרגיל "מעשה בחמישה בלונים" **מהספר מבוא לתכנות בסביבת אינטרנט, מבוא ו-HTML**, פרק 6.

א. כל מעבר עכבר מעל צבע של בלון יהפוך את הרקע לאותו צבע ואת שם הצבע ללבן. כאשר העכבר יוסט מצבע הבלון, יחזרו הצבעים לקדמותם. במילים אחרות: כאשר יופעל הקובץ, יופיע הטקסט בצבע שחור על רקע לבן, פרט לשמות הבלונים המוצגים בצבע הבלון. כאשר, לדוגמה, יהיה סמן העכבר מעל המילה כחול (שצבעה כחול על רקע לבן) הרקע ישתנה לכחול, והמילה כחול תופיע בצבע לבן. כאשר העכבר יוזז מהבלון הכחול, צבע הרקע יחזור להיות לבן והמילה כחול תופיע בצבע כחול.

ב. בצע שינוי זה רק בהופעה הראשונה של כל צבע בטקסט.

שאלה 5.2

כתוב מסמך ובו הפיסקה: "כך אמר יוסף טרומפלדור לפני מותו:". בלחיצה על הפיסקה יופיע: "אין דבר, טוב למות בעד ארצנו".

חומר עזר

עוד על אירועים תוכל לקרוא בספר **JavaScript סדנת לימוד**, בהוצאת **הוד-עמי**, פרק 25.

פונקציות מחזירות ערך

כשפונקציה מסיימת את פעולתה היא מחזירה ערך כלשהו. השימוש בפונקציות הוא כדי לבצע שינוי בערך משתנה ולקבל ערך בחזרה, או כדי לבצע פעולה כלשהי ולהחזיר ערך תוצאה. ב-JavaScript, כפי שראית, ניתן לבנות את רוב הקוד במסגרת הפונקציה. נניח שבנית פונקציה שמוציאה שורש למספר וצריכה להציג את התוצאה. את כל הקוד תוכל לכתוב במסגרת הפונקציה ואין לך צורך לכתוב משפטים אחרים מחוץ לפונקציה כדי לבצעה במלואה.

יש מקרים בהם יש צורך להשתמש ביכולת של הפונקציה להחזיר ערך. למשל, אתה מתכנן פונקציה שתקבל מספר ותחזיר תשובה כלשהי אם המספר גדול שווה לאפס ותשובה אחרת אם המספר קטן מאפס.

נבקש לקבוע את שורת הסטטוס על ידי המשפט הבא:

```
window.status = '2' ;
```

כתוצאה, נקבל בשורת הסטטוס את המחרוזת '2'. אולם, ניתן לקבוע ערך דינמי על ידי קביעת הערך שיוחזר מפונקציה. לדוגמה:

```
window.status = sayHello() ;
```

אז, ייקבע ערך שורת הסטטוס על פי הערך שיוחזר מהפונקציה sayHello(). ניתן להשתמש ב-JavaScript במילה השמורה return. פקודה זו מסיימת את פעולת הפונקציה ו"יוצאת" ממנה, וגם מחזירה את הערך שרשום אחריה. לדוגמה:

```
<script language="javascript">
   function sayHello()
   {
    return "Hello World!" ;
   }
</script>
```

בדוגמה זו הפונקציה תחזיר את המחרוזת "Hello World!". בדוגמה **5-04.html** ניתן לראות שלחיצה על הלחצן קובעת את ערך שורת הסטטוס על ידי קריאה לפונקציה sayHello().

דוגמה 5-04.html:

```
<html>
 <head>
  <title>5-04</title>
  <script language="javascript">
   function sayHello()
   {
    return "Hello World!" ;
   }
```

```
  </script>
 </head>
<body>
  <form name="butForm">
   <input type="button"
    value="לחץ כאן והתבונן בשורת המצב"
    onclick="window.status=sayHello();" />
   <input type="button"
    value="לחץ כאן לניקוי שורת המצב"
    onclick="window.status='';" />
   </form>
 </body>
</html>
```

בדוגמה הקודמת הפונקציה החזירה מחרוזת, אך היא יכולה גם להחזיר ערך של משתנה, כפי שניתן לראות בדוגמה הבאה.

דוגמה 5-05.html:

```
<html dir="rtl">
 <head>
  <title>5-05</title>
  <script language="javascript">
   function click2show()
   {
    var x ;
    x = 2 ;
    return x ;
   }
  </script>
 </head>
 <body>
  <form>
   <input type="button"
    value="לחץ כאן. מה אתה רואה בשורת המצב?"
    onclick="window.status=click2show();" />
  </form>
 </body>
</html>
```

כל אחד יכול לבחור לטעמו את דרך הקריאה לפונקציה.

טיפ

במשפט return אפשר להחזיר גם true או false, במיוחד במקרים שבהם מתבצעת הפונקציה כדי לתת אור "ירוק" או אור "אדום" (עובר/לא עובר, נכון/לא נכון).

שאלה 5.3

בנה דף המכיל פיסקה משירה של רחל המשוררת. תן לפיסקה מזהה (id) בדרך הבאה:

```
<div id="ConText">
רק על עצמי לספר ידעתי<br />
צר עולמי כעולם נמלה<br />
גם משאי עמסתי כמוה<br />
רב וכבד מכתפי הדלה<br />
</div>
```

גודל הגופן באלמנט המוגדר תחת התגית מוגדר כך:

```
document.all.ConText.style.fontSize = 12 ;
```

צבע הגופן באלמנט המוגדר תחת התגית מוגדר כך:

```
document.all.ConText.style.color = 'red' ;
```

א. עשה שבמעבר על הפיסקה גודל הגופן ישתנה ל-38, וביציאה מהפיסקה גודל הגופן יהיה גודלו הרגיל.

ב. לחיצה על הפיסקה תשנה את הצבע לאדום.

ג. הוסף מונה שיאפשר לבצע הפעלתם של 5 אירועים בלבד בדף זה.

חומר עזר

עוד על פונקציות תוכל לקרוא בספר **JavaScript סדנת לימוד**, בהוצאת **הוד-עמי**, פרק 18.

פרויקט סיינפלד

בשלב זה של הלימוד כבר עשית חזרה על התחביר של JavaScript. זוהי חזרה בלבד כי תכנות, בשלב זה, אתה אמור לדעת. JavaScript היא שפת תכנות והתחביר שלה דומה לתחביר של Pascal, C, ++C, Visual Basic או Java. יש הבדלים בין שפות התכנות למיניהן, אבל ברמה הבסיסית משתנה זה משתנה ולולאת for זאת לולאת for ומשפט if יש בכל שפה, כך שפרקים 1 עד 4 כולל נועדו להזכיר לך מבנים תחבירים ואת אופן כתיבתם בשפת JavaScript.

בפרק 1 הכרת את המושג אובייקט (object), שיטה (method) ותכונה (property). בפרק 2 ו-3 הכרת את האובייקט document שהוא אובייקט המסמך (מסמך HTML שלך). השיטה ()write של האובייקט document מציגה את הפלט כחלק מאובייקט המסמך, זה שנראה בחלון. השיטה ()alert שייכת לאובייקט אחר בשם window. אם היית רוצה לדייק היית צריך לכתוב ()window.alert, אבל את שמו של אובייקט window נוכל להשמיט (זה חלק מגמישות השפה לטוב ולרע). אובייקט המסמך, זה שקראנו לו document, הוא חלק מחלון הדפדפן, כלומר הוא חלק מאובייקט window. לכן, כדי להפעיל את השיטה ()write של אובייקט document שהוא חלק מאובייקט window, צריך לכתוב ()window.document.write - ארוך אבל מדויק.

בפרק 3 הכרת את השיטה bgColor של האובייקט document שאם רוצים להתייחס אליה צריך לרשום window.document.bgColor. את הערך המתקבל תוכל להציג בתיבת דו-שיח על ידי השיטה ()alert, כך:

```
window.alert(window.document.bgColor) ;
```

ואתה כבר יודע שאפשר שלא לציין את שמו של אובייקט window ואת המשפט הקודם לרשום כך:

```
alert(document.bgColor) ;
```

וזה בדיוק אותו דבר.

את הערך של window.document.bgColor נוכל גם להציב במשתנה לצורך המשך עבודה. כמובן שנוכל לשנות את ערכה של התכונה באופן הבא:

```
window.document.bgColor = "red" ;
```

פרק 4 הינו חזרה על תחביר הלולאות ומשפטי הבקרה שאתה כבר מכיר משפות תכנות אחרות.

המשימה היא לשנות את צבע הרקע של דף **about.html** כאשר הסמן עובר מעל לטקסט **אודות** שמוצג. לשם כך נשתמש באירועים onmouseover ו- onmouseout באופן הבא:

```
<h2 onmouseover="document.bgColor='#ABAF33';"
    onmouseout ="document.bgColor='#FBE995';">
    <font color="navy">
      אודות
    </font>
</h2>
```

אתה יכול לשער מה יקרה אם תכתוב document.bg**c**olor במקום document.bg**C**olor? נסה ותהנה...

ניתן היה גם לכתוב:

```
<h2>
    <font color="navy"
       onmouseover="document.bgColor='#ABAF33';"
       onmouseout ="document.bgColor='#FBE995';">
      אודות
    </font>
</h2>
```

שים לב, האובייקט הנוצר על ידי התגית <h2> משתרע מצד ימין של המסמך ועד לצידו השמאלי (לכל רוחבו של החלון). במקרה זה, האירועים ייטענו כשהסמן יעבור בשורה בה רשום הטקסט **אודות** ולא רק שהסמן ממש על הטקסט **אודות**.

כדי שצבע הרקע ישתנה רק כאשר הסמן עובר מעל לטקסט **אודות**, יש לכתוב:

```
<h2>
    <font color="navy">
      <span  onmouseover="document.bgColor='#ABAF33';"
             onmouseout ="document.bgColor='#FBE995';">
        אודות
      </span>
    </font>
</h2>
```

עבור לחלון **החברות של ג'רי**. הפעם לחיצה על הטקסט **החברות של ג'רי** תשנה את צבע הרקע של המסך וגם את צבע הגופן. בכל מקרה שלא הגדרת צבע לגופן, יוגדר צבע עבורך. אם לא תגדיר את המאפיין bgcolor בתגית <body> הוא יהיה לבן (#FFFFFF). אם לא תגדיר את המאפיין text במסגרת התגית <body> הוא יהיה שחור (#000000). המאפיין bgcolor בתגית <body> הוא התכונה bgColor באובייקט document. המאפיין text בתגית <body> הוא התכונה fgColor באובייקט document.

להלן המשימה בפרטי פרטים:

- לחיצה בודדת על הטקסט **החברות של ג'רי** תשנה את צבע הרקע ל- #FE7070 ואת צבע הגופן ל- #1E90FF.
- לחיצה כפולה על הטקסט **החברות של ג'רי** תשנה את צבע הרקע ל- #8B4513 ואת צבע הגופן ל- #FF6347.
- הזזת העכבר מחוץ לטקסט **החברות של ג'רי** תחזיר את הצבעים לקדמותם.

הפתרון:

נתחיל מזה שאתה מעוניין לשמור את צבע הרקע המקורי של הדף document.bgColor ואת צבע הגופן המקורי של הדף document.fgColor. ההזדמנות לבצע זאת היא בעת טעינת הדף, כלומר בעת שהאירוע onload יופעל במסגרת התגית <body>. לשם כך בנה פונקציה בשם originalColor():

```
<script language="JavaScript">
   var originalBgColor ;
   var originalFgColor ;

   function originalColor()  {
     originalBgColor = document.bgColor ;
     originalFgColor = document.fgColor ;
   }

</script>
```

שים לב ששני המשתנים הוגדרו מחוץ לפונקציה כדי שיהיו משתנים גלובליים. אם המשתנים היו מוגדרים בתוך הפונקציה originalColor(), רק פונקציה זו (שבה הם מוגדרים) היתה מכירה בהם.

את הפונקציה originalColor() תפעיל מתוך התגית <body> בעת טעינת המסמך:

```
<body bgcolor="#FBE995" onload="originalColor();">
```

את הטקסט **החברות של ג'רי** הקף בתגית עם האירועים onclick, ondblclick ו- onmouseout באופן הבא:

```
<span onclick = "document.bgColor = '#FE7070';
                 document.fgColor = '#1E90FF';"
      ondblclick = "document.bgColor = '#8B4513';
                    document.fgColor = '#FF6347';"
      onmouseout = "back2original();">
  החברות של ג'רי
</span>
```

שים לב לפונקציה back2original() שמפעיל האירוע onmouseout. פונקציה זו תחזיר למסמך את צבעיו המקוריים.

להלן הקוד המלא שבדף:

```
<html dir="rtl">
  <head>
    <title>Girls</title>

    <script language="JavaScript">
       var originalBgColor ;
       var originalFgColor ;

       function originalColor()  {
         originalBgColor = document.bgColor ;
         originalFgColor = document.fgColor ;
       }

       function back2original()  {
         document.bgColor = originalBgColor ;
         document.fgColor = originalFgColor ;
       }

    </script>

  </head>
<body bgcolor="#FBE995" onload="originalColor();">
<h2>
  <font color="navy">
    <span onclick = "document.bgColor = '#FE7070';
                     document.fgColor = '#1E90FF';"
          ondblclick = "document.bgColor = '#8B4513';
                        document.fgColor = '#FF6347';"
          onmouseout = "back2original();">
      החברות של ג'רי
    </span>
  </font>
</h2>
```

פרק 6

תמונות, מערכים ואנימציה

הוספת תמונה למסמך HTML נעשית באמצעות התגית <img />. כדי לבצע שינויים בתמונות שבמסמך HTML יש צורך להתייחס לתמונה בשם המזהה אותה, כלומר לתת ערך למאפיין name של התגית <img />. התגית <img /> יוצרת אובייקט מסוג Image. כלומר, כאשר אנו משתמשים בתגית **<img />**, למעשה אנו אומרים לדפדפן להכין מקום עבור התמונה (תוכל לנסות זאת על ידי ציון התגית <img /> ללא המאפיין src וראה מה מציג הדפדפן). ניצור מקום לתמונה וניתן לה שם:

```
<img name="myPic" />
```

כעת, כאשר יש מקום לתמונה במסמך וגם שם (name) המציין מקום זה בצורה חד-ערכית, המאפיינים של התגית <img /> מאפשרים לקבוע, לדוגמה, את הקובץ שיופיע במקום זה. הכן שתי תמונות: אחת בשם yellowPlane.gif ושנייה בשם bluePlane.gif והשתמש באחת מהן.

```
<img name="myPic" src="images/yellowPlane.gif" />
```

הערה

אין חובה להשתמש דווקא בקובץ תמונה בפורמט gif. דפדפן IE מכיר פורמטים אחרים כמו jpg, png וגם bmp.

מכיון שמיקום התמונה קבוע, ניתן לפנות אליו דרך תסריט JavaScript ולשנות את ערך התכונה src. להזכירך, המאפיין src בתגית <img /> והתכונה src באובייקט מסוג Image מציינים את שם הקובץ שתוכנו יוצג.

שאלה ותשובה!

מדוע src הוא מאפיין ופעם src הוא תכונה?

כאשר src מוגדר במסגרת התגית <img /> הוא נקרא מאפיין. התגית <img /> יצרה אובייקט מסוג Image שאחת מתכונותיו היא src. הערך של התכונה src באובייקט Image הוא הערך שניתן למאפיין src בתגית <img />.

בדוגמה **6-01.html**, יש דוגמה לתסריט המחליף תמונה כאשר העכבר עובר מעל התמונה. לצורך כך יש ליצור מקום לתמונה ולמקם בה קובץ תמונה. בנוסף, יש להוסיף event-handler בשם **onmouseover**.

```
<img name="myPic" src="images/yellowPlane.gif" border="0"
   ↳ onmouseover="picChange();" />
```

כאשר תעביר את סמן העכבר מעל אובייקט התמונה, ייטען event-handler בשם onmouseover שיקרא לפונקציה ()picChange שמייד תיצור. פונקציה זו תפנה אל מיקום התמונה ותשנה את ערך התכונה src של אובייקט התמונה. כדי לפנות למיקום תמונה בדף, יש לציין את שם מיקום התמונה (במקרה זה myPic) שהיא אובייקט של ה**מסמך** (document), כלומר document.myPic. לאחר ציון שם התמונה ניתן לקבוע את ערך התכונה src.

```
<script language="javascript">
   function picChange()
   {
     document.myPic.src = "images/bluePlane.gif" ;
   }
</script>
```

נסה בעצמך את הדוגמה **6-01.html**.

חומר עזר

פירוט כל השיטות והתכונות של אובייקט Image תוכל למצוא בספר **JavaScript סדנת לימוד** בהוצאת **הוד-עמי**, פרק 27 ובנספח ב', ובספר **JavaScript המדריך השלם** בהוצאת **הוד-עמי**.

דוגמה 6-01.html:

```
<html>
 <head>
  <script language="javascript">
   function picChange()
   {
    document.myPic.src = "images/bluePlane.gif" ;
   }
  </script>
 </head>
 <body>
  <img name="myPic" src="images/yellowPlane.gif" border="0"
    onmouseover="picChange();" />
 </body>
</html>
```

כלומר, במעבר העכבר על התמונה יופעל event-handler בשם onmouseover שיקרא לפונקציה ()picChange, אשר קובעת ערך חדש לתכונה src של אובייקט myPic שהוא אובייקט מסוג Image.

לאובייקט שהוגדר ב-DOM ניתן לגשת במספר דרכים. למשל, כדי לגשת לתכונה src של אובייקט מסוג תמונה myPic רשמת:

```
document.myPic.src = "images/bluePlane.gif" ;
```

תוכל גם לרשום:

```
document['myPic'].src = "images/bluePlane.gif" ;
```

וגם זה בסדר:

```
window.document['myPic'].src = "images/bluePlane.gif" ;
```

אתה זוכר שהאובייקט document הוא אובייקט-בן של אובייקט window. לאובייקט window, כמו לכל אובייקט, יש **אוספים** (collections), **שיטות** (methods), **תכונות** (properties), **אובייקטים בנים** וגם **event-handlers**.

שמות אובייקטים

כדי שתוכל לגשת לתמונה באמצעות מודל האובייקטים (DOM - Document Object Model), יהיה עליך לתת לתגית <img /> שם באמצעות המאפיין name או המאפיין id של התגית. תוכל לתת שם לתגית <img />, <form>, , וגם <table>, <frame /> ועוד הרבה תגיות אחרות. כל תגית שרשמת ב-HTML היא אובייקט ב-DOM. התגית <img /> מגדירה אובייקט מסוג Image, שהוא שונה מאובייקט מסוג Link הנוצר מהתגית <a>, למשל. לכל אובייקט יש את האוספים, השיטות והתכונות שלו.

לא חובה לתת ערך למאפיין name או id של התגית, אבל אם אתה מעוניין לגשת לאובייקט כדי לשנות את תכונותיו בעזרת תסריט JavaScript, יהיה עליך לתת לו שם.

הערה

לדפדפן IE מספיק לתת שם בעזרת המאפיין id. המאפיין name הינו שארית מדפדפן Netscape שכאמור לא מכיר במאפיין id. ב-IE גם המאפיין id יעבוד וגם המאפיין name.

באופן מעשי, לתגיות ו- משתמשים במאפיין id ולתגיות כמו <form>, <a>, <img /> ואחרות משתמשים במאפיין name.

בכתיבת שמות יש לנהוג לפי כללים מקובלים:

- התו הראשון חייב להיות אות (a..z, A..Z).
- שם יכול להיות מחרוזת תווים, החל באות אחת ועד עשרות אותיות. מותר להשתמש באותיות האלף בית הלועזיות (a..z, A..Z), בספרות (0..9) ובשני תווים מיוחדים: _ ו-$.
- אסור להשתמש **ברווח** (space character).

מומלץ לקבוע שמות בעלי משמעות המתארים את מהות האובייקט. בנושא זה התפתחו מספר שיטות למתן שמות שמטרתם אחת: להפוך את התוכנית שלך קלה יותר לכתיבה ולקריאה. אני לא מתנגד לחלוטין לקרוא לאובייקט בשמות כמו i, j או a, אבל אפשר גם אחרת.

השיטה ההונגרית (Hungarian convention) מוסיפה קידומת לפני שם האובייקט המציינת את סוגו. לאובייקט Image הוסף את הקידומת img או pic, למשל: imgDog, imgHomePage, picLogo. לאובייקט Link (שנוצר על ידי התגית <a>) הקידומת יכולה להיות lnk (לא ink), למשל: lnkUpperFrame, lnkStartPage. קידומות נוספות: chk לאובייקט checkbox, txt לאובייקט text, frm לאובייקט form וכדומה.

בשיטה הנקראת formated compound words כל אות ראשונה היא אות ראשית, למשל: MaxiumWeight ו- FindHeightInMeters.

בווריאציה של השיטה שלעיל, המילה הראשונה תמיד באותיות קטנות ואחר כך כל אות ראשונה במילה היא אות ראשית, למשל: maxiumWeight ו- findHeightInMeters.

ויש גם המפרידים בין מילה למילה בקו תחתי, למשל: Maxium_Weight ו- Find_Height_In_Meters.

תן לאובייקטים שמות בשפה האנגלית, כמו myPicture ולא HatmunaSeli, מכיון שגם לך יהיה קשה לקרוא את הקוד שכתבת ולא רק לאחרים. שים לב שהשמות לא יהיו "ארוכים" מדי. למרות שאין מגבלה לאורך שם האובייקט, נסה לצמצם ולתת שם שתוכל לעבוד איתו ולהבין את תפקידו בתוכנית. במקום אובייקט בשם txtUserPerviosPicture רשום txtUserPrevPic, במקום picMenuMessage רשום picMenuMsg.

לא חשוב איזו שיטה תבחר, העיקר שתהיה עקבי.

הערה

כדי שלא תבלבל את עצמך, תן לאובייקטים שמות שאינם שמות של מילים שמורות ו/או של אוספים, שיטות או תכונות. זה לא יהיה נבון לתת לאובייקט את השם bgcolor כאשר יש תכונה בשם זאת. זה רק עשוי לבלבל.

העברת נתונים לפונקציה

בדרך שהודגמה לעיל היה צורך לבנות פונקציה נפרדת עבור כל החלפת תמונה. זה בסדר לשתיים או שלוש החלפות, אבל מה לעשות כאשר צריך לבצע החלפות רבות? האם צריך לבנות פונקציה עבור כל החלפה? לא. ניתן לבנות פונקציה מודולארית המקבלת נתונים בעת הקריאה אליה. כלומר, כל הפניות יתבצעו לפונקציה אחת שתדע לטפל במיקומים שונים וקבצי תמונות שונים.

העברת הנתונים מתבצעת בקריאה עצמה בין הסוגריים, כלומר:

```
<img name="plane" src="images/yellowPlane.gif" border="0"
 onmouseover="changeMyPic('plane','images/greenPlane.gif');" />
```

הנתונים שנבחר להעביר הם מיקום התמונה (זהו שם אובייקט התמונה) שאת התכונה שלו ברצוננו לשנות (במקרה זה - plane) ואת הקובץ הגרפי שברצוננו למקם במקום זה (במקרה זה images/greenPlane.gif).

כלומר, במעבר עכבר תתבצע קריאה לפונקציה changeMyPic() ויישלחו אליה שני נתונים: plane (שם אובייקט) ו- image/greenPlane.gif (הנתיב המלא לקובץ התמונה).

על הפונקציה לקבל את הנתונים. לכן, יש להכין משתנים שיכילו את הנתונים שנשלחו. המשתנים החדשים נרשמים בתוך הסוגריים. כדאי לתת למשתנים שמות פונקציונליים, כגון picLocation ו- picFileName:

```
function changeMyPic(picLocation, picFileName)
```

כעת, ניתן להחליף את ההוראות הקבועות בפונקציה במשתנים שיצרנו:

```
document[picLocation].src = picFileName ;
```

שים לב לאופן הגישה לאובייקט. לא ניתן לגשת לאובייקט כך: document.picLocation.src, מכיון שלא קיים אובייקט picLocation. כדי לגשת לאובייקט בשם הערך של משתנה picLocation, יש לרשום document[picLocation].src. מכיון שהאובייקט בשם plane הוא חלק מהאוסף (collection) של כל התמונות במסמך, ניתן לגשת לאובייקט מסוג תמונה דרך האוסף images, באופן הבא:

```
document.images[picLocation].src = picFileName ;
```

כעת, ברור לך ולכל מי שיתבונן בתוכנית שלך שמדובר באובייקט מסוג תמונה, כי הוא חלק מהאוסף images.

חבר את כל פיסות הקוד שבנית, **דוגמה 6-02.html**:

```
<html>
 <head>
  <title>6-02</title>
  <script language="javascript">
   function changeMyPic(picLocation, picFileName)
   {
    document.images[picLocation].src = picFileName ;
   }
  </script>
 </head>
 <body>
  <img name="plane" src="images/yellowPlane.gif" border="0"
   onmouseover="changeMyPic('plane','images/greenPlane.gif');" />
  </body>
</html>
```

כאשר ה**פונקציה מודולרית**, ניתן לפנות אליה בבקשת שינוי כל תמונה על המסך. להזכירך, המונח "מודולריות" מתייחס לקטע קוד שאינו קבוע, אלא משתנה ונקבע במהלך ריצת התוכנית.

את האובייקט plane תוכל לחשוף לתסריט JavaScript באחת הדרכים הבאות:

```
document.images['plane'].
document['plane'].
plane.
```

זה היופי שב-DOM.

נשתמש במודולריות הפונקציה, כדי לקרוא לה גם בעת יציאת העכבר, כלומר onmouseout. **דוגמה 6-03.html**:

```
<html>
 <head>
  <title>6-03</title>
  <script language="javascript">
   function switchPic(picLoc, picFileName)
   {
    document.images[picLoc].src = picFileName ;
   }
  </script>
 </head>
 <body>
  <img name="aircraft" src="images/greenPlane.gif" border="0"
   onmouseover="switchPic('aircraft','images/bluePlane.gif');"
   onmouseout="switchPic('aircraft','images/greenPlane.gif');" />
 </body>
</html>
```

בדוגמה **6-03.html** מתבצעת קריאה לפונקציה ()switchPic גם עם יציאת העכבר מעל התמונה (onmouseout), אך הנתונים שנשלחים הם שונים במקצת. הנתון הראשון הוא אותו נתון, מכיון שברצוננו להחזיר את התמונה המקורית לאותו מיקום. הנתון השני קובע את הקובץ אותו ברצוננו להציג במיקום.

דרך קלה יותר להעברת האובייקט היא בעזרת המילה השמורה this. כאשר אתה מעביר/שולח לפונקציה את המילה this, אתה מעביר את האובייקט בו נלכד האירוע. במקום לכתוב:

```
<img name="aircraft"
 src="images/greenPlane.gif"
 border="0"
 onmouseover="switchPic('aircraft','images/bluePlane.gif');"
 onmouseout="switchPic('aircraft','images/greenPlane.gif');" />
```

תוכל לרשום:

```
<img name="aircraft"
 src="images/greenPlane.gif"
 border="0"
 onmouseover="switchPic(this,'images/bluePlane.gif');"
 onmouseout="switchPic(this,'images/greenPlane.gif');" />
```

כאשר אתה מעביר לפונקציה ()switchPic את this, אתה מעביר את האובייקט window.document.aicraft, מדליק אהה?

שאלה ותשובה!

למה השימוש ב-this עדיף על השימוש בשם האובייקט?

זה עניין של נוחות. מה עדיף לך? את זה אתה תקבע.

החלפת תמונות במעבר עכבר

ניתן לבצע החלפה פשוטה של תמונה בעת שסמן העכבר עובר מעליה, תוך שימוש בקוד משובץ, בצורה הבאה:

דוגמה 6-04.html:

```
<html>
 <head>
  <title>6-04</title>
 </head>
 <body>
  <img name="airPlane"
    src="images/bluePlane.gif" border="0"
    onmouseover="document.airPlane.src='images/yellowPlane.gif';"
    onmouseout="document.airPlane.src='images/bluePlane.gif';" />
 </body>
</html>
```

כלומר, ניתן לקבוע את ערך src של מיקום התמונה באותו מקום שבו ממוקם ה- event-handler.

אובייקט תמונה, טעינה מוקדמת

הרעיון ביצירת אובייקט Image לכל תמונה שתרצה להציג הוא לבצע טעינה מוקדמת של התמונה לזיכרון המחשב. כאשר התמונה מוחלפת באמצעות קוד הנמצא בתגית, היא נטענת למחשב ברגע שהאירוע (event-handler) מתרחש. כאשר נוצר אובייקט תמונה, התמונה נטענת לזיכרון המחשב עוד לפני הצגתה. ברגע שמתרחש האירוע שמפעיל את החלפת התמונה, היא מוצגת מיידית כי היא נמצאת בזיכרון המחשב המקומי. אובייקט תמונה הוא מסוג **Image** והתחביר ליצירתו הוא:

```
newPic = new Image() ;
```

המילה new מאתחלת את האובייקט. כלומר, newPic נוצר כמשתנה אובייקט מ**סוג תמונה** ולכן ניתן לפנות לתכונותיו, כלומר אל newPic.src. לדוגמה:

```
<script language="javascript">
    pic1 = new Image() ;
    pic1.src = "images/greenPlane.gif" ;
    pic2 = new Image() ;
    pic2.src = "images/bluePlane.gif" ;
</script>
```

כדי שהטעינה המוקדמת תעבוד, יש לבצע אותה לפני כל דבר אחר בדף. המשמעות היא שיש לכתוב את התסריט שיבצע את הטעינה המוקדמת במסגרת התגית <head>. אם התסריט של הטעינה המוקדמת נכתב במסגרת פונקציה, יש להפעילה באמצעות האירוע onload בתגית <body>.

חומר עזר

פירוט כל השיטות והתכונות של אובייקט Image תוכל למצוא בספר **JavaScript סדנת לימוד** בהוצאת **הוד-עמי**, פרק 26 ובנספח ב', ובספר **JavaScript המדריך השלם** בהוצאת **הוד-עמי**.

שאלה 6.1

סדרת התמונות הבאה משמשת להצגת 5 לחצנים (את התמונות תוכל למצוא באתר האינטרנט של הספר בכתובת: www.hod-ami.co.il/59355.html). בהתחלה מוצגים כל הלחצנים (תמונות) שהאות האחרונה בשם הקובץ היא L. כאשר העכבר נמצא על הלחצן, מתחלפת התמונה של הלחצן לאותו שם קובץ עם סיומת R. לדוגמה, כאשר הסמן יהיה על התמונה internetL.gif היא תתחלף לתמונה internetR.gif.

<picture>L.gif	**<picture>R.gif**
internetL.gif	internetR.gif
progL.gif	progR.gif
mcseL.gif	mcseR.gif
pctechL.gif	pctechR.gif
office2kL.gif	office2kR.gif

א. בנה דף ובו חמישה לחצנים המורכבים מתמונות מתחלפות על ידי מתן ערך לתכונה המתאימה במסגרת התגית.

ב. פתור את השאלה מהסעיף הקודם עם חמשת הלחצנים והפעם על ידי בניית דף בו התמונות יוחלפו על ידי העברת ערכים לשתי פונקציות.

מערך

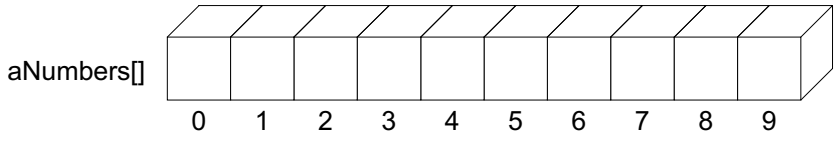
מערך (array) הוא אחד הכלים רבי העוצמה העומדים לרשות תוכניתנים ברוב שפות התכנות. המערך הוא למעשה טבלה המכילה שורות ועמודות. מערך המכיל שורה אחת (מספר העמודות גדול שווה 1) נקרא מערך חד-מימדי. מערך המכיל יותר משורה אחת (מספר העמודות גדול מ-1) נקרא דו-מימדי. המערך מיועד לאחסן סדרה של נתונים, כמו: תוצאות מבחן של תלמידים בכיתה, שמות הסטודנטים בשנה א', רשימת פריטים שבמחסן ועוד. בדרך כלל, המערך מכיל סדרה של נתונים כמו: שמות, ציונים, טמפרטורה, ולכן כולם באותו סוג נתונים - כל הנתונים במערך הם טקסט או כולם מספר וכדומה. כך זה בשפת C. ב-JavaScript איברי המערך יכולים להכיל נתונים מסוגים שונים (איבר אחד מכיל מחרוזת טקסט, איבר אחר, באותו מערך, מכיל מספר). אם כן, במערך מאוחסנים נתונים רבים תחת שם אחד ואפשר לגשת אליהם על פי מספר סידורי.

הגדרה והצבה

המערך הפשוט, שבו יש **ערך אחד** בלבד בכל איבר, הוא מערך **חד-מימדי**. לדוגמה, אם יש לאחסן שמות תלמידים בכיתה, תוכל ליצור מערך שלצורך הדוגמה ייקרא students (אתה כבר יודע שתוכל לבחור באיזה שם שתחפוץ). את המערך יוצרים על ידי שימוש במילה השמורה new והגדרת המשתנה student כמשתנה מסוג מערך, על ידי שימוש במילה השמורה Array() (שים לב לסוגריים הצמודים).

הגדרת המשתנה students כמשתנה מסוג מערך נראית כך:

```
var students ;
students = new Array() ;
```



תרשים 6.1: מערך חד-מימדי בשם []aNumbers המכיל 10 משתנים (ריקים בינתיים).

ניתן להציב ערכים במערך במספר דרכים:

הצבה תוך כדי הגדרה:

```
var students ;
students = new Array("rami", "einat", "david", "rina") ;
```

האיבר הראשון במערך students יכיל את הערך rami, האיבר השני במערך students יכיל את הערך einat, האיבר השלישי במערך students יכיל את הערך david והאיבר הרביעי במערך students יכיל את הערך rina.

במערך, מספרו הסידורי של האיבר הראשון הוא 0, מספרו הסידורי של האיבר השני הוא 1 וכך הלאה. כך שבדוגמה שלעיל, האיבר הראשון במערך students המכיל את הערך rami נמצא במקום סידורי 0, האיבר השני במערך students המכיל את הערך einat נמצא במקום סידורי 1, וכך הלאה.

הצבה:

```
var students ;
students = new Array() ;
students[0] = "rami" ;
students[1] = "einat" ;
students[2] = "david" ;
students[3] = "rina" ;
```

השימוש בסוגריים המרובעים מאפשר גישה לערכים ספציפיים במערך על פי המספר הסידורי שלהם. כדי לא להתבלבל, אני מציע להשאיר את האיבר הראשון (זה שבמקום אפס) ריק, אז תוכל לומר שהאיבר במקום הראשון נמצא באיבר סידורי מספר 1. לצורך כך יש לכתוב את ההצבה למערך בדרך הבאה:

```
var students ;
students = new Array("", "rami", "einat", "david", "rina");
```

או כך:

```
var students ;
students = new Array() ;
students[1] = "rami" ;
students[2] = "einat" ;
students[3] = "david" ;
students[4] = "rina" ;
```

נסה את **דוגמה 6-05.html**:

```
<html dir="rtl">
 <head>
  <title>6-05</title>
  <script language="javascript">
   var i ;
   var students ;

   students = new Array() ;
   students[1] = "rami" ;
   students[2] = "einat" ;
   students[3] = "david" ;
   students[4] = "rina" ;
```

```
   for (i = 1 ;  i <= 4 ;  i++) {
       document.write("ערכו של איבר מספר " + i + " = " + students[i]) ;
       document.write("<br />") ;
   }
  </script>
 </head>
<body>
</body>
</html>
```

הדרך השנייה להצבת ערכים במערך, היא על ידי קביעתם עבור כל איבר לפי מספרו הסידורי. כלומר, לאחר הגדרת המשתנה students כמשתנה מסוג Array, תוכל לבצע הצבה של כל אחד מהערכים על פי מספר האיבר שלו, כגון:

```
 students[1] = "rami" ;
```

בדוגמה **6-06.html**, הקובץ יוצר בדרך שונה את המערך שראינו בדוגמה **6-05.html**.

דוגמה 6-06.html:

```
<html dir="rtl">
 <head>
  <title>6-06</title>
  <script language="javascript">
   var i ;
   var students ;
students = new Array("" ,"רינה" ,"דוד" ,"עינת" ,"רמי") ;

   for (i = 1 ;  i <= 4 ;  i++) {
     document.write("ערכו של איבר מספר " + i + " = " + students[i]) ;
     document.write("<br />") ;
   }
  </script>
 </head>
<body>
</body>
</html>
```

שים לב שבהצבת ערכים בעברית במערך תוך כדי הגדרה, סדר השדות משתבש. בגלל סיבה זאת וכדי למנוע טעויות או אי-הבנות בהצבת ערכים למערך, אני מעדיף את השיטה בה כל ערך מוצב לאיבר סידורי יחיד, כך:

```
var students ;
students = new Array() ;
students[1] = "רמי" ;
students[2] = "עינת" ;
students[3] = "דוד" ;
students[4] = "רינה" ;
```

בשיטת הצבה זאת, ברור לאיזה איבר יוצב איזה ערך.

יתרונות המערך

כאשר פועלים על נתונים במערך, ניתן להשתמש במספר הסידורי של האיברים כדי להריץ לולאות על כל האיברים או על חלק מהם, כפי שדרוש. בדוגמה הבאה יש לבדוק אם התלמידה **עינת** נמצאת בין תלמידי הכיתה. מספר הפעמים שהלולאה תפעל יהיה כמספר איברי המערך. זכור! במערך כמו זה שבדוגמה **6-06.html** יש 5 איברים! החל ממקום 0 ועד מקום 4, סה"כ 5 איברים. מספר איברי המערך נמצא בתכונה **length**, כלומר students.length שערכו בדוגמה זו הוא 5. תחביר הלולאה ייראה כך:

```
for (i = 0 ;  i <= students.length-1 ;  i++)
```

בכל ריצה של הלולאה נבדוק את הערך התורן במערך על פי המספר הסידורי i, אשר ישתנה במהלך ביצוע התוכנית מ- 0 ועד 4. מכיון שבקובץ שבדוגמה **6-06.html** לא נעשה שימוש באיבר במיקום 0, משתנה הלולאה צריך להיות מ- 1 ועד 4, כך:

```
for (i = 1 ;  i <= students.length-1 ;  i++)
```

פקודת הבדיקה היא:

```
if (students[i] == "עינת")
```

חשוב לציין שערכי מערכים יכולים להיות **מחרוזות**, **מספרים** או **ערכים בוליאנים** (true/false). בתוכנית שבדוגמה **6-07.html** מתבצעת בדיקה פשוטה לצורך הדגמה של השימוש במספר הסידורי של איברי המערך. זוהי שיטה יעילה על מערך קטן ולא ממוין, ולכן לא מומלץ להשתמש בלוגיקה פשוטה זו במערכים גדולים.

דוגמה 6-07.html:

```
<html dir="rtl">
 <head>
  <title>6-07</title>
  <script language="javascript">
   var i
   var students ;

   students = new Array("" ,"רינה" ,"דוד" ,"עינת" ,"רמי") ;
   studentName = "עינת" ;
   for (i = 1 ;  i <= students.length-1 ;  i++) {
    if (students[i] == studentName) {
     document.write("מצאתי את  " + studentName + " במיקום " + i) ;
     break ;
    }
   }
  </script>
 </head>
<body>
</body>
</html>
```

שיטות הקשורות למערכים

- **join()** - השיטה join() מאפשרת צירוף ערכי המערך למחרוזת אחת ארוכה. כלומר, אם נשתמש במערך students ונרצה להציג את כל ערכיו ללא שימוש בלולאה, נכתוב:

```
document.write(students.join()) ;
```

על המסך יוצג:

```
,rami,einat,david,rina
```

או

```
,רמי,עינת,דוד,רינה
```

שים לב: התו הראשון שיוצג באיברי המערך הוא פסיק (,), מכיון שקבענו שהאיבר הראשון במערך (זה שמיקומו 0) לא יכיל דבר. למעשה, התא הזה אינו מוגדר.

- **sort()** - השיטה sort() משמשת למיון ערכי המערך. אם ערכי המערך הם מחרוזות טקסט, השיטה sort() תסדר את איברי המערך לפי סדר אלפביתי. כלומר, אם ניצור את המערך letters בצורה הזו:

```
var letters ;
letters = new Array("c", "a", "b", "e", "d") ;
```

נוכל לסדר את האיברים לפי סדר אלפביתי על ידי שימוש בשיטה sort(), הנה כך:

```
letters.sort() ;
document.write(letters.join()) ;
```

על המסך תופיע המחרוזת: a,b,c,d,e

אם איברי המערך היו מכילים מספרים, התוצאה היתה סידור של הערכים בסדר מיון מספרי עולה.

- **reverse()** - השיטה reverse() הופכת את סדר המערך. כלומר, מערך המכיל את הערכים x,a,k לפי סדר זה יהפוך לאחר שימוש בשיטה reverse() למערך המכיל את אותם ערכים בסדר הפוך: k,a,x.

חומר עזר

פירוט כל השיטות והתכונות של אובייקט Array תוכל למצוא בספר **JavaScript סדנת לימוד** בהוצאת **הוד-עמי**, פרק 19, 20 ובנספח א', ובספר **JavaScript המדריך השלם** בהוצאת **הוד-עמי**.

שאלה 6.2

		נכון	לא נכון	הסבר
1	var Shula = new Array ;			
2	HeartsDesire = Array() ;			
3	new espectador = Array () ;			
4	var new Array(latinMusic) ;			
5	var RiverDeep = new Array();			
6	surfaceID = new Array();			

סמן את המשפטים הלא נכונים בטבלה שלעיל והסבר.

אנימציה

כדי לייצר אפקט של אנימציה, יש להציג מספר תמונות זו אחר זו בהפרשי זמן. הכן מספר תמונות מסוג gif או jpg לצורך ביצוע אנימציה.

אחת הדרכים הנוחות לעבוד עם מספר רב של עצמים היא דרך מערך (array).

שלב ראשון בבניית הקוד שיריץ אנימציה הוא יצירת אובייקט התמונה. מכיון שבדוגמה זו נעשה שימוש בשלוש תמונות, יהיה עליך ליצור שלושה אובייקטים:

```
<script language="javascript">
       onePic = new Image() ;
       onePIC.src = "images/geenPlane.gif" ;
       twoPic = new Image() ;
       twoPic.src = "images/yelowPlane.gif" ;
       threePic = new Image() ;
       threePic.src = "images/bluePlane.gif" ;
```

זה הזמן לבנות מערך (array) שיכיל את ערך התכונה src של האובייקטים של התמונות:

```
pic = new Array() ;
pic[0] = onePic.src ;
pic[1] = twoPic.src ;
pic[2] = threePic.src ;
```

כפי ששמת לב, עדיין לא כתבת פונקציה ולכן כל המשתנים שיצרת ושתיצור מחוץ לפונקציות יהיו למעשה משתנים **גלובליים.** משתנים **גלובליים** הם משתנים אשר ניתן להתייחס אליהם מכל פונקציה, בניגוד למשתנים **מקומיים** אשר נוצרים בתוך פונקציה וניתן לגשת אליהם רק מהפונקציה שבה נוצרו.

צריך משתנה שישמש כאינדקס ובעזרתו תציג תמונה אחר תמונה לפי הסדר. מכיון שנעשה שימוש ב- 3 תמונות שה-URL שלהן נמצא במערך ממקום 0 עד 2, צריך משתנה **גלובלי** שערכיו יהיו 0, 1, 2, 0, 1, 2 וחוזר חלילה.

```
var picIndex = 0 ;
```

בשלב זה יש ליצור את הפונקציה. הפונקציה תקבע את ערך src של אובייקט התמונה airCraft, אשר תיצור בהמשך באמצעות התגית <img />. בעזרת המשתנה הגלובלי picIndex, תעבור על פני מערך pic ממקום 0 עד 2 וחוזר חלילה. הערך ההתחלתי של המשתנה הגלובלי picIndex יהיה 0.

```
function animation()
 {
    document.images['airCraft'].src = pic[picIndex] ;
    picIndex++ ;
```

שים לב כי לאחר קביעת הערך, ערכו של picIndex גדל באחד. כך, בפעם הבאה שהפונקציה תופעל, יהיה ערך src של התמונה: pic[1].

כדי למנוע מצב בו יגדל ערכו של picIndex מעל למספר האיברים (אובייקטים) במערך, יש להוסיף את התנאי הבא שתפקידו לחסום את הלולאה:

```
if (picIndex > 2)  picIndex = 0 ;
```

כעת, יש לדאוג שהפונקציה תפעל שוב ושוב. נבצע זאת בעזרת השיטה **setInterval()**.

שאלה 6.3

השיטה Math.random() מחזירה מספר שווה או גדול מ- 0 וקטן מ- 1. בנה תוכנית לבחירת 20 מספרים אקראיים ושלמים בין 0 ל- 65. בעזרת שיטה זו הצב אותם במערך.

חומר עזר

פירוט כל השיטות והתכונות של אובייקט Math תוכל למצוא בספר **JavaScript סדנת לימוד** בהוצאת **הוד-עמי**, פרק 6 ובנספח א', ובספר **JavaScript המדריך השלם** בהוצאת **הוד-עמי**.

שאלה 6.4

בנה תוכנית שבמעבר אחד על איברי המערך תמצא את המספר הגדול ביותר במערך ואת המספר הקטן ביותר במערך ואת מיקומם.

השיטה setInterval()

פונקציה מתבצעת פעם אחת כשקוראים לה. כדי להפעיל פונקציה יותר מפעם אחת, צריך לקרוא לה מתוך לולאה. כאשר הקריאה לפונקציה נעשית מתוך לולאה (וזה לא משנה אם הלולאה מתבצעת פעם אחת, עשר או אלף פעמים), לא ניתן לבצע כל פעולה אחרת. המשמעות היא, שבעת שהלולאה הופעלה, לחיצה על קישור או על לחצן לא תתקבל על ידי הדפדפן כי הוא "עסוק". באמצעות השיטה **setInterval()** ניתן ליצור מעין תהליך עיבוד מקביל של הפעלת פונקציה בפרקי זמן קבועים מראש:

```
setInterval('alert("Hello");',1000) ;
```

הפונקציה setInterval() שבדוגמה תמתין 1000 אלפיות שנייה ואז תבצע alert(), תמתין עוד 1000 אלפיות שנייה ואז תבצע alert(), תמתין עוד 1000 אלפיות שנייה... וכך עד שתופעל פקודת עצירה בשם clearInterval().

כדי שניתן יהיה לעצור את ההפעלה החוזרת של הפונקציה שהופעלה על ידי setInterval(), יש לאתחל את setInterval() בתוך משתנה, כמו בדוגמה זו:

```
x = setInterval('animation();',1000) ;
```

כאשר השיטה נמצאת בתוך משתנה, ניתן לבטל אותה על ידי שימוש בשיטה **clearInterval()** בצורה הבאה:

```
clearInterval(x) ;
```

אם הפונקציה clearInterval() תופעל על משתנה שלא הוכרז, תופיע הודעת שגיאה. כדי למנוע מצב בו תופעל הפונקציה clearInterval(x) לפני שהופעלה הפונקציה setInterval() באמצעות משפט הצבה למשתנה x, יש להצהיר על x כמשתנה גלובלי.

כל שנותר הוא ליצור את התמונה ב-HTML על ידי שימוש בתגית img />> ולקרוא לפונקציה initAnimation(), שבעזרת הפונקציה setInterval() מפעילה את הפונקציה animation(). בקוד שבהמשך, הקריאה מתבצעת על ידי event-handler בשם onload שנמצא בתגית <body>.

לפניך הקוד המלא לביצוע האנימציה.

דוגמה 6-08.html:

```
<html>
 <head>
  <title>6-05</title>
  <script language="javascript">
   onePic = new Image() ;
   onePic.src = "images/greenPlane.gif" ;
   twoPic = new Image() ;
   twoPic.src = "images/bluePlane.gif" ;
   threePic = new Image() ;
```

```
    threePic.src = "images/yellowPlane.gif" ;

    pic = new Array() ;
    pic[0] = onePic.src ;
    pic[1] = twoPic.src ;
    pic[2] = threePic.src ;

    var picIndex ;
    picIndex = 0 ;
    var x ;

    function initAnimation() {
     x = setInterval('animation();',1000) ;
    }

    function animation() {
     document.images['airCraft'].src = pic[picIndex] ;
     picIndex++ ;
     if (picIndex > 2) picIndex = 0 ;
    }
   </script>
  </head>
  <body onload="initAnimation();">
     <img name="airCraft" src="images/yellowPlane.gif" />
  </body>
</html>
```

שאלה 6.5

סדרת התמונות הבאה מתארת את צי מטוסי אל-על (התמונות נמצאות בקבצים של פרק זה הניתנים להורדה מהאתר הספר באינטרנט בכתובת www.hod-ami.co.il/59355.html):

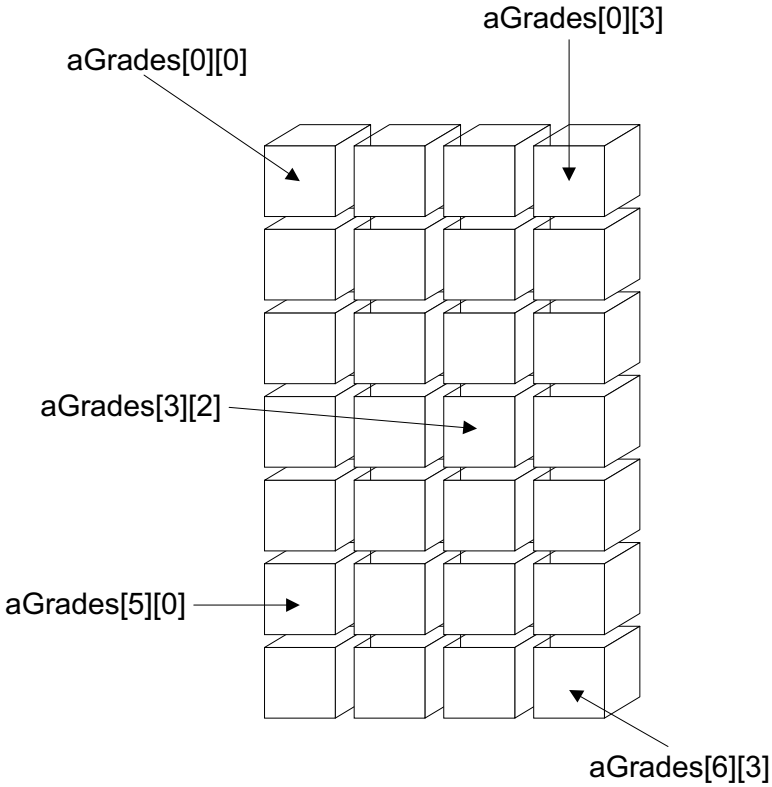
737-700.jpg
737-800.jpg
747-200.jpg
747-400.jpg
757-200.jpg
767-200.jpg

א. בנה דף המכיל שני לחצנים: לחצן אחד שיציג את תמונות המטוסים זו אחר זו (החל מהתמונה הראשונה עד האחרונה וחוזר חלילה - 1, 2, 3, 4, 5, 6, 1, 2, 3...) ולחצן נוסף שיעצור את הצגת התמונות.

ב. שנה את סדר הצגת התמונות כך שיוצגו בסדר: 1, 2, 3, 4, 5, 6, 5, 4, 3, 2, 1, 2, 3... וחוזר חלילה.

מערכים רב-מימדיים

במערך רגיל יש ערך אחד בכל אחד מאיברי המערך. ניתן להעצים את המערך על ידי יצירת מערך דו-מימדי. מערך דו-מימדי דומה לטבלה שבה יש יותר מעמודה אחת. בכל אחת מהעמודות יש ערכים מאותו סוג. לדוגמה, כדי לייצג לוח שחמט נצטרך לבנות מערך דו-מימדי שיכיל את כל המשבצות בלוח. מערך המכיל את הפרטים האישיים של כל העובדים גם הוא מערך דו-מימדי ששורותיו מייצגות את העובדים השונים, וכותרתו מייצגת את הפרטים שאנו שומרים עבור כל אחד מהם.



תרשים 6.2: דוגמה למערך דו-מימדי בשם aGrades[] עם 7 שורות ו- 4 עמודות.

בדוגמה הבאה יוצג מערך דו-מימדי פשוט - טבלה המורכבת משורות ועמודות. הטבלה מייצגת לדוגמה, בניין 4 קומות הממוספרות: 0, 1, 2, 3 ובכל קומה שלוש דירות הממוספרות: 0, 1, 2. להלן הערכים שיהיו באיברי המערך:

A1	A2	A3
B1	B2	B3
C1	C2	C3
D1	D2	D3

להלן התהליך הגדרת מערך דו-מימדי:

הגדרת מערך חד-מימדי:

```
vData = new Array() ;
```

הגדרת כל אחד מאיברי המערך החד-מימדי כמערך חד-מימדי:

```
vData[i] = new Array() ;
```

כאשר i מייצג את מספר השורות במערך.

הצבת ערכים במערך דו-מימדי:

```
vData[i][j] = ערך ;
```

כאשר i מייצג את מספר השורה ו- j מייצג את מספר העמודה.

בבעיה העומדת לפניך יש להגדיר מערך דו-מימדי של 4 שורות ו- 3 עמודות, כך ש-i יקבל ערכים 0, 1, 2 או 3 ו-j יקבל ערכים 0, 1 או 2.

נציב את הערכים הנדרשים במערך הדו-מימדי, כך:

```
vData = new Array() ;

vData[0] = new Array() ;
   vData[0][0] = "A1" ;
   vData[0][1] = "A2" ;
   vData[0][2] = "A3" ;

vData[1] = new Array() ;
   vData[1][0] = "B1" ;
   vData[1][1] = "B2" ;
   vData[1][2] = "B3" ;

vData[2] = new Array() ;
   vData[2][0] = "C1" ;
   vData[2][1] = "C2" ;
   vData[2][2] = "C3" ;
```

```
vData[3] = new Array() ;
   vData[3][0] = "D1" ;
   vData[3][1] = "D2" ;
   vData[3][2] = "D3" ;
```

לאחר הגדרת המערך ואתחול איבריו בערכים, יהיה עליך ליצור לולאה שתבנה טבלת HTML בעלת ארבע שורות ושלושה תאים בכל שורה. מכיון שטבלה היא למעשה ייצוג של מערך דו-מימדי, נציג את המערך vData[][] בתוך תאי הטבלה. לצורך כך, נבנה שתי לולאות מקוננות, אשר הראשונה תבצע ארבע ריצות והמונה שלה i ייצג את המימד הראשון של המערך, כלומר vData[i]. לולאה זו גם תתחיל ותסיים את שורות הטבלה. הלולאה הפנימית תרוץ שלוש פעמים, תבנה את תאי הטבלה ותציג בכל תא את אחד מאיברי המערך. מונה הלולאה הפנימית j ייצג את איברי המימד השני, כלומר תת-המערכים vData[i][j]:

```
document.write('<table border="1" cellpadding="30">') ;

for (i = 0 ;  i < 4 ;  i++)
{
 document.write('<tr>') ;
 for(j = 0 ;  j < 3 ;  j++)
 {
  document.write('<td>') ;
  document.write(vData[i][j]) ;
  document.write('</td>') ;
 }
 document.write('</tr>')
}
document.write('</table>')
```

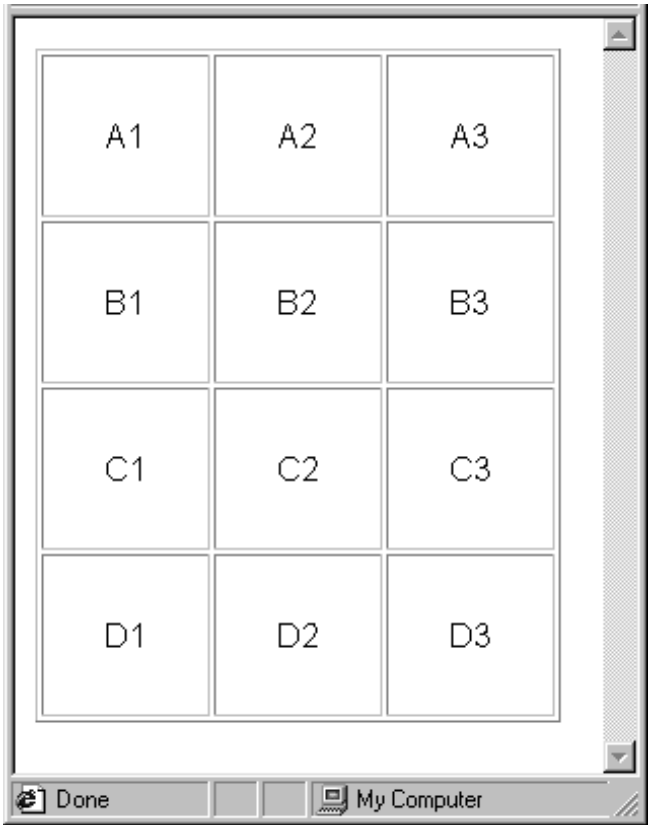
כמובן, יש לפתוח ולסגור את הטבלה בצורה מסודרת. לפניך הקוד המלא.

דוגמה 6-09.html:

```
<html>
 <head>
  <title>6-09</title>
  <script language="javascript">
   vData = new Array() ;

   vData[0] = new Array() ;
   vData[0][0] = "A1" ;
   vData[0][1] = "A2" ;
   vData[0][2] = "A3" ;
```

```
   vData[1] = new Array() ;
   vData[1][0] = "B1" ;
   vData[1][1] = "B2" ;
   vData[1][2] = "B3" ;

   vData[2] = new Array() ;
   vData[2][0] = "C1" ;
   vData[2][1] = "C2" ;
   vData[2][2] = "C3" ;

   vData[3] = new Array() ;
   vData[3][0] = "D1" ;
   vData[3][1] = "D2" ;
   vData[3][2] = "D3" ;

   document.write('<table border="1" cellpadding="30">') ;

   for (var i = 0 ;  i < 4 ;  i++)
   {
    document.write('<tr>') ;
    for(var j = 0 ;  j < 3 ;  j++)
    {
     document.write('<td>') ;
     document.write(vData[i][j]) ;
     document.write('</td>') ;
    }
    document.write('</tr>')
   }
   document.write('</table>')
  </script>
 </head>
<body>
</body>
</html>
```

זו תהיה התוצאה בדפדפן:



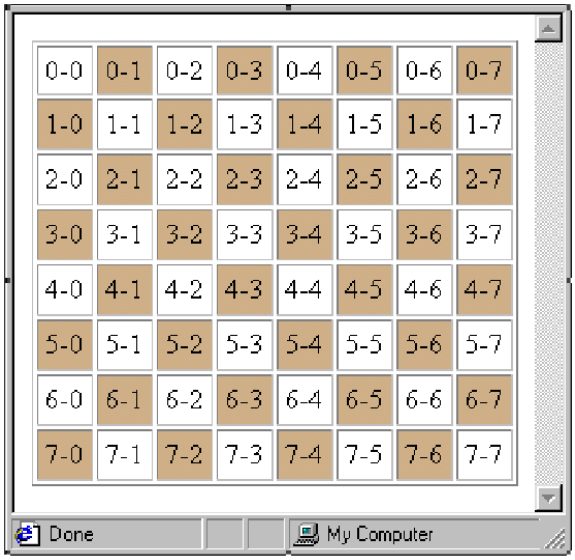
תרשים 6.3: יצירת מסמך HTML חדש בעזרת תסריט.

הדוגמה הבאה משתמשת בשתי הלולאות המקוננות ליצירת המערך הדו-מימדי ולהצגתו בטבלה. התוצאה הסופית היא מבנה לוח שחמט. כדאי לעקוב אחר ערכי המשתנים בכל מחזור של הלולאה.

דוגמה 6-10.html :

```
<html>
 <head>
  <title>6-10</title>
  <script language="javascript">
   var tmp ;
   document.write('<table width="100%" height="100%"
    ↳ border="1">') ;
   chessBoard = new Array() ;
   for(var i = 0 ;  i < 8 ;  i++)
   {
    tmp = i % 2 ;
    chessBoard[i] = new Array() ;
    document.write('<tr>') ;
    for(var j = 0 ;  j < 8 ;  j++)
    {
     tmp++ ;
     if (tmp % 2 == 0) {
      colorName = 'tan' ;
     }else{
      colorName = 'white' ;
     }
     chessBoard[i][j] = i + "-" + j ;
     document.write("<td bgcolor=") ;
     document.write("'" + colorName + "'") ;
     document.write(" align='center'>") ;
     document.write(chessBoard[i][j]) ;
     document.write("</td>") ;
    }
    document.write('</tr>')
   }
   document.write('</table>')
  </script>
 </head>
<body>
</body>
</html>
```

התוצאה בדפדפן תיראה כך:



תרשים 6.4: שימוש בלולאת for ומשפט if.

שאלה 6.6

משחק הזיכרון הוא משחק קלפים בו יש מספר כלשהו של זוגות של תמונות (זהות או משלימות). בנה את משחק הזיכרון לפי הכללים הבאים:

א. מטרת המשחק לגלות את כל זוגות התמונות.

ב. במשחק 6 תמונות (3 זוגות זהים).

ג. המשחק מיועד לשחקן אחד.

ד. התמונות מסודרות ב- 3 שורות. בכל שורה 2 תמונות.

ה. מיקום של כל תמונה נקבע באופן אקראי.

ו. התמונות מוצגות כלפי מטה.

ז. יש לבחור שתי תמונות אחת אחרי השנייה.

ח. אם שתי התמונות זהות, אז המשחק ממשיך לגילוי הזוג הבא.

ט. אם התמונות אינן זהות, אז התמונות מתהפכות והמשחק ממשיך.

י. המשחק מסתיים כשהמשתמש מגלה את כל הזוגות.

פרויקט סיינפלד

רשימת הקישורים במסגרת מימין, דף menu.html, מורכבת מטקסט. בעזרת תוכנה גרפית תוכל ליצור שתי סדרות של לחצנים. לכל קישור יהיו שני לחצנים: לחצן אחד שיוצג שהסמן לא עליו ולחצן שני שיוצג כשהסמן עליו. השוני בין שני הלחצנים יכול להיות צבע, גודל, תזוזה או כל שוני ויזואלי אחר.

לדוגמה, הנה הלחצנים שאני יצרתי. את הלחצנים ניתן להוריד מאתר הספר באינטרנט בכתובת: www.hod-ami.co.il/59355.html):

קישור	סמן מחוץ ללחצן	סמן על הלחצן
דף הבית	דף הבית	דף הבית
אודות	אודות	אודות
החברות של ג'רי	החברות של ג'רי	החברות של ג'רי
מועדון	מועדון	מועדון
חידון	חידון	חידון
מילון	מילון	מילון

שם לחצן בסדרת הלחצנים כשהסמן מחוץ ללחצן יתחיל בטקסט offButton, ושם לחצן בסדרה כאשר הסמן על הלחצן יתחיל בטקסט onButton. להלן שמות הלחצנים השונים בהתאמה לטבלה שלעיל:

קישור	שם הדף	סמן מחוץ ללחצן	סמן על הלחצן
דף הבית	HomePage.html	offButtonHomePage.gif	onButtonHomePage.gif
אודות	About.html	offButtonAbout.gif	onButtonAbout.gif
החברות של ג'רי	Girls.html	offButtonGirls.gif	onButtonGirls.gif
מועדון	Club.html	offButtonClub.gif	onButtonClub.gif
חידון	Trivia.html	offButtonTrivia.gif	onButtonTrivia.gif
מילון	Yada.html	offButtonYada.gif	onButtonYada.gif

השיטתיות במתן השמות לתמונות רק יכולה לעזור: הקישור לדף Yada.html יוצג על ידי שני לחצנים offButtonYada.gif ו- onButtonYada.gif, כך שברור לכל מי שמתבונן בקוד מי שייך למי.

בתוכנית זו תהיה רק פונקציה אחת שתשנה את התמונה. פונקציה זו צריכה לדעת את שם התמונה ואת שמו של האובייקט בו יתבצע החילוף. כלומר, לפונקציה צריך להעביר שני נתונים (פרמטרים).

להלן הגדרה של הלחצן המקשר לדף הבית:

```
<a href="homePage.html" title="דף הבית" target="content">
 <img src="images/offButtonHomePage.gif" border="0"
  onmouseover = "changeButtonPic(this, 'images/onButtonHomePage.gif');"
  onmouseout = "changeButtonPic(this, 'images/offButtonHomePage.gif');">
</a>
```

שים לב לפרמטרים שמועברים לפונקציה changeButtonPic(). הפרמטר הראשון הוא המילה השמורה this. בפרמטר זה עובר האובייקט בו נלכד האירוע לפונקציה. הפרמטר השני הוא כתובת התמונה שתוחלף.

וכך נראית הפונקציה changeButtonPic():

```
<script language="JavaScript">

  function changeButtonPic(buttonObj, newPic)  {
    buttonObj.src = newPic ;
  }

</script>
```

הפונקציה changeButtonPic() קולטת את האובייקט בו נלכד האירוע במשתנה that (יכולתי כמובן לקרוא לו בכל שם אחר) ואת כתובת התמונה במשתנה newPic. הפונקציה changeButtonPic() משנה למעשה את ערך התכונה (property) של האובייקט בו נלכד האירוע שזו התגית <img />.

באותו אופן יש להמשיך ולעדכן את שאר הקישורים בדף menu.html. וכך הוא נראה:

```
<html dir="rtl">
  <head>
    <title>Menu</title>

  <script language="JavaScript">

  function changeButtonPic(buttonObj, newPic)  {
    buttonObj.src = newPic ;
  }
  </script>

  </head>
<body bgcolor="#F1DF9F">
<a href="homePage.html" title="דף הבית" target="content">
```

```
 <img src="images/offButtonHomePage.gif" border="0"
   onmouseover = "changeButtonPic(this, 'images/onButtonHomePage.gif');"
   onmouseout = "changeButtonPic(this, 'images/offButtonHomePage.gif');" />
</a>
<br />
<a href="about.html" title="אודות האתר" target="content">
 <img src="images/offButtonAbout.gif" border="0"
   onmouseover = "changeButtonPic(this, 'images/onButtonAbout.gif');"
   onmouseout = "changeButtonPic(this, 'images/offButtonAbout.gif');" />
</a>
<br />
<a href="Girls.html" title="החברות של ג'רי" target="content">
 <img src="images/offButtonGirls.gif" border="0"
   onmouseover = "changeButtonPic(this, 'images/onButtonGirls.gif');"
   onmouseout = "changeButtonPic(this, 'images/offButtonGirls.gif');" />
</a>
<br />
<a href="Club.html" title="מועדון החברים של סיינפלד" target="content">
 <img src="images/offButtonClub.gif" border="0"
   onmouseover = "changeButtonPic(this, 'images/onButtonClub.gif');"
   onmouseout = "changeButtonPic(this, 'images/offButtonClub.gif');" />
</a>
<br />
<a href="Trivia.html" title="אתה חושב שאתה יודע הכל?" target="content">
 <img src="images/offButtonTrivia.gif" border="0"
   onmouseover = "changeButtonPic(this, 'images/onButtonTrivia.gif');"
   onmouseout = "changeButtonPic(this, 'images/offButtonTrivia.gif');" />
</a>
<br />
<a href="Yada.html" title="Yada, Yada, Yada" target="content">
 <img src="images/offButtonYada.gif" border="0"
   onmouseover = "changeButtonPic(this, 'images/onButtonYada.gif');"
   onmouseout = "changeButtonPic(this, 'images/offButtonYada.gif');" />
</a>
<br />
<a href="mailto:info@seinfeld.co.il" title="שלח דואר אלקטרוני למפעילי
האתר" target="content">
  צור קשר
</a>
<br />
<p>
<font size="-2">
  600 האתר מומלץ לצפיה ברזולוציה x800
</font>
```

```
</p>
</body>
</html>
```

אם בגלל התמונות משתבשת התצוגה, יש לשנות את גודל המסגרות בקובץ SeinfeldFrame.html.

העשרה

הקטע הבא הוא מעט יותר מורכב ואני יותר רואה אותו כהעשרה ולא כחומר חובה.

הבעיה בתפריט הלחצנים שיצרנו הוא שלא נוכל לדעת, על פי מבט בלחצנים, איזה מהם נבחר אחרון. הלוגיקה היתה שתמונת הלחצן משתנה כאשר העכבר מעל האובייקט (תמונה) או כאשר העכבר לא מעל האובייקט (תמונה), ולכן לא ניתן "לשמר" את הלחצן במצב לחוץ.

נגדיר משתנה גלובלי בשם iClickThisButtonNAME, שבו נציב את שם התמונה שהסמן היה עליה בעת לחיצה, כפי שהוגדר במאפיין name בתגית <img />. במשתנה גלובלי בשם iClickThisButonSRC נציב את הכתובת של התמונה בלחצן שנלחץ. לצורך כך יש להגדיר את המאפיין name במסגרת התגית <img />. נוסיף הפעלה של פונקציה בעת שהאירוע onclick מתרחש.

בעת הפעלת הפונקציה changeButtonPic(), יהיה עלינו לבדוק את ערכו של משתנה iClickThisButtonNAME לפני שינוי התמונה.

להלן קוד המקור של דף menuADV.html:

```
<html dir="rtl">
  <head>
    <title>Menu</title>

  <script language="JavaScript">

  var iClickThisButtonNAME ;
  var iClickThisButtonSRC ;
  iClickThisButtonNAME = '' ;

  function changeButtonPic(buttonObj, newPic)  {
    if (iClickThisButtonNAME != buttonObj.name)
      buttonObj.src = newPic ;
  }

  function clickButton(buttonObj, newPic)  {
   if (iClickThisButtonNAME != '')  {
    if (iClickThisButtonNAME != buttonObj.name) {
     iClickThisButtonNAME = "'" + iClickThisButtonNAME + "'" ;
     document.all[eval(iClickThisButtonNAME)].src = iClickThisButtonSRC ;
      }
    }
    iClickThisButtonNAME = buttonObj.name ;
    iClickThisButtonSRC = buttonObj.src ;
    buttonObj.src = newPic ;
  }

  </script>

  </head>
<body bgcolor="#F1DF9F">
<a href="homePage.html" title="דף הבית" target="content">
 <img src="images/offButtonHomePage.gif" border="0"
   name="homepage"
   onmouseover = "changeButtonPic(this, 'images/onButtonHomePage.gif');"
   onmouseout = "changeButtonPic(this, 'images/offButtonHomePage.gif');"
   onclick = "clickButton(this, 'images/onButtonHomePage.gif');" />
</a>
<br />
```

```
<a href="about.html" title="אודות האתר" target="content">
 <img src="images/offButtonAbout.gif" border="0"
   name="about"
   onmouseover = "changeButtonPic(this, 'images/onButtonAbout.gif');"
   onmouseout = "changeButtonPic(this, 'images/offButtonAbout.gif');"
   onclick = "clickButton(this, 'images/onButtonAbout.gif');" />
</a>
<br />
<a href="Girls.html" title="החברות של ג'רי" target="content">
 <img src="images/offButtonGirls.gif" border="0"
   name="girls"
   onmouseover = "changeButtonPic(this, 'images/onButtonGirls.gif');"
   onmouseout = "changeButtonPic(this, 'images/offButtonGirls.gif');"
   onclick = "clickButton(this, 'images/onButtonGirls.gif');" />
</a>
<br />
<a href="Club.html" title="מועדון החברים של סיינפלד" target="content">
 <img src="images/offButtonClub.gif" border="0"
   name="club"
   onmouseover = "changeButtonPic(this, 'images/onButtonClub.gif');"
   onmouseout = "changeButtonPic(this, 'images/offButtonClub.gif');"
   onclick = "clickButton(this, 'images/onButtonClub.gif');" />
</a>
<br />
<a href="Trivia.html" title="אתה חושב שאתה יודע הכל?" target="content">
 <img src="images/offButtonTrivia.gif" border="0"
   name="trivia"
   onmouseover = "changeButtonPic(this, 'images/onButtonTrivia.gif');"
   onmouseout = "changeButtonPic(this, 'images/offButtonTrivia.gif');"
   onclick = "clickButton(this, 'images/onButtonTrivia.gif');" />
</a>
<br />
<a href="Yada.html" title="Yada, Yada, Yada" target="content">
 <img src="images/offButtonYada.gif" border="0"
   name="yada"
   onmouseover = "changeButtonPic(this, 'images/onButtonYada.gif');"
   onmouseout = "changeButtonPic(this, 'images/offButtonYada.gif');"
   onclick = "clickButton(this, 'images/onButtonYada.gif');" />
</a>
<br />
<a href="mailto:info@seinfeld.co.il" title="שלח דואר אלקטרוני למפעילי
האתר" target="content">
צור קשר
</a>
```

```
<br />
<p>
<font size="-2">
  600 האתר מומלץ לצפיה ברזולוציה 800x
</font>
</p>
</body>
</html>
```

הבעיה עדיין לא נפתרה. הלחצן לא חוזר למצבו. הסיבה היא שבעת לחיצה על הלחצן התמונה שלו היא onButton בעוד אנו רוצים להחזיר אותו למצב offButton. המשך בפרק הבא.

פרק 7

אובייקט המחרוזת

אחד היתרונות הגדולים של JavaScript הוא היכולת לבצע בדיקה ועיבוד קלט המוזן על ידי המשתמש בטופס HTML. פרטי הטופס נבדקים במחשב הלקוח בלי צורך לבצע גישה למחשב השרת. כדי לטפל בטקסט אנו משתמשים באובייקט **String**.

אובייקט המחרוזת - String Object

כדי ליצור משתנה המכיל מחרוזת טקסט צריך רק לאתחל משתנה ולהציב לתוכו ערך של מחרוזת טקסט. משתנה מחרוזת הוא למעשה אובייקט מסוג **String**, וכעת, ניתן ליהנות גם מהשיטות והתכונות הכלולות בו.

ניצור משתנה ונעניק לו ערך של מחרוזת טקסט:

```
var strMyLady ;
strMyLady = "Anna Nicole Smith" ;
```

מבחינת הדפדפן, המשתנה strMyLady הוא אובייקט מסוג מחרוזת (String), ולכן ניתן להשתמש בתכונה length אשר מחזירה את אורך המחרוזת שהוא מספר התווים המופיעים בה, קובץ **AnnaNicoleSmith.html**.

```
<html>
 <head>
  <title>Anna Nicole Smith</title>
  <script langauge="javascript">
   var strMyLady ;
   strMyLady = "Anna Nicole Smith" ;
   document.write("<b>" + strMyLady + "</b>") ;
   document.write(" length = " + strMyLady.length) ;
   window.alert (strMyLady.length) ;
  </script>
 </head>
<body>
</body>
</html>
```

בדוגמה שלעיל נעשה שימוש בתכונה length של האובייקט strMyLady, כדי להציג את המחרוזת על ידי document.write() וגם על ידי window.alert().
לביטוי strMyLady.length יש ערך מתמטי (במקרה זה 17) ולכן ניתן לבצע איתו חישובים, למשל:

```
myGirl = strMyLady.length + 22 ;
```

חומר עזר

פירוט כל השיטות והתכונות של אובייקט String תוכל למצוא בספר **JavaScript סדנת לימוד** בהוצאת **הוד-עמי**, פרק 21 ונספח א', ובספר **JavaScript המדריך השלם** בהוצאת **הוד-עמי**.

טיפול בנתוני טופס

בדיקה ראשונית לטקסט שהוזן על ידי המשתמש

כדי לאפשר למשתמש להזין נתונים בדף HTML, משתמשים ב**טפסים** (forms). כדי לאפשר לשפת JavaScript לגשת אל שדות הקלט, יש להגדיר שם לטופס ושם לכל אחד מהשדות הכלולים בו. לדוגמה:

```
<body>
 <form name="myForm">
   הקלד את פרטיך האישיים:
   <hr />
   שם פרטי:
   <input type="text" name="firstName" />
   <br />
   שם משפחה
   <input type="text" name="lastName" />
   <hr />
   לחץ על הלחצן
   <input type="button" value="השווה" onclick="checkUp();" />
   כדי לראות איזה שם יותר ארוך
 </form>
</body>
```

בדוגמה זו יש שני שדות טקסט ולחצן המכיל אירוע (event-handler) בשם **onclick**, אשר בעת לחיצה על הלחצן יפעיל את הפונקציה checkUp(). הפונקציה checkUp() בודקת איזה מהשמות (שם פרטי ושם משפחה) ארוך יותר. לצורך כך, על הפונקציה לברר את מספר התווים של הערך שהוקלד אל תוך השדות. זיהוי השדה נעשה כך:

באופן כללי:

```
window.document.form.text
```

שם פרטי:

```
window.document.myForm.firstName
```

שם משפחה:

```
window.document.myForm.lastName
```

כלומר, אובייקט השדה (במקרה זה אובייקט מסוג text) הוא תת-אובייקט של אובייקט הטופס (אובייקט מסוג form), וזה בתורו מהווה תת-אובייקט לאובייקט המסמך (document), שהוא תת-אובייקט של אובייקט החלון (window). ה-DOM מאפשר גישה ישירה לאובייקט הטופס בלי לציין את אובייקט החלון ואובייקט המסמך.

באופן כללי:

```
form.text
```

שם פרטי:

```
myForm.firstName
```

שם משפחה:

```
myForm.lastName
```

אם תרצה, תוכל גם לגשת לטופס דרך האוסף (collection) הנקרא forms. הטופס הראשון במסמך נמצא במקום 0 באוסף:

```
window.document.forms[0].
window.document.forms['myForm'].
```

חומר עזר

פירוט כל השיטות והתכונות של אובייקט Form תוכל למצוא בספר **JavaScript סדנת לימוד** בהוצאת **הוד-עמי**, פרק 24, 25, 26 ונספח ב', ובספר **JavaScript המדריך השלם** בהוצאת **הוד-עמי**.

אין צורך להוסיף את אובייקט ה**מסמך** (document), אולם כדאי לעשות זאת מטעמים של נוהל כתיבה נכון:

באופן כללי:

```
document.form.text
```

שם פרטי:

```
document.myForm.firstName
```

שם משפחה:

```
document.myForm.lastName
```

ערך שדה הטקסט נמצא בתכונה value:

```
document.myForm.firstName.value
```

ערך השדה יוצב במשתנה, וכך נוצר אובייקט מחרוזת:

```
strFirstName = document.myForm.firstName.value ;
strLastName  = document.myForm.lastName.value ;
```

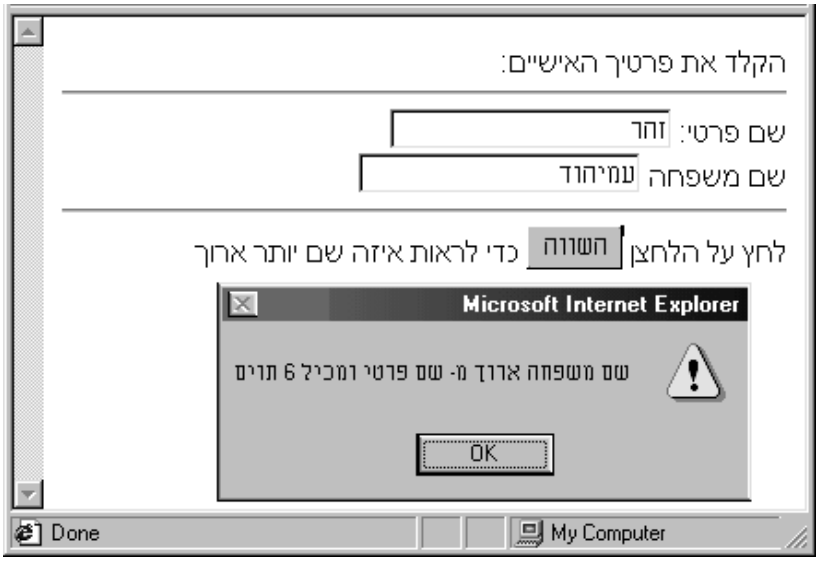
המשתנים strFirstName ו-strLastName מכילים את ערכי שדות הטקסט, ולכן הם אובייקטים מסוג **אובייקט מחרוזת** (String Object). כך ניתן לבדוק את התכונה length שלהם וגם להשוות בין שניהם, למשל.

```
function checkUp()
{
 var Msg ;
 var strFirstName ;
 var strLastName ;

 Msg = "" ;
 strFirstName = document.myForm.firstName.value ;
 strLastName  = document.myForm.lastName.value ;

 if (strFirstName.length != strLastName.length) {
  if (strFirstName.length > strLastName.length) {
   Msg = "שם פרטי ארוך מ- שם משפחה ומכיל "  ;
   Msg = Msg + strFirstName.length ;
   Msg = Msg + " תווים" ;
  }else{
   Msg = "שם משפחה ארוך מ- שם פרטי ומכיל " ;
   Msg = Msg + strLastName.length ;
   Msg = Msg + " תווים" ;
  }
```

```
 }else{
   Msg = "שם פרטי ו- שם משפחה באותו אורך של " ;
   Msg = Msg + strLastName.length ;
   Msg = Msg + " תווים)" ;
 }
 alert(Msg) ;
}
```

שים לב למשתנה Msg המשרשר את הודעת הפלט.

ראה **דוגמה 7-01.html**.



תרשים 7.1 : השוואה בין אורך (מספר תווים) של הערכים שבשדות טופס.

חומר עזר

פירוט כל השיטות והתכונות של אובייקט Text תוכל למצוא בספר **JavaScript סדנת לימוד** בהוצאת **הוד-עמי**, נספח ב', ובספר **JavaScript המדריך השלם** בהוצאת **הוד-עמי**.

ערך של שדה טופס הוא תמיד מחרוזת תווים. האובייקט Global מכיל מספר שיטות מעניינות וביניהן השיטה parseInt() המחזירה את הערך המספרי השלם של המחרוזת בשדה, והשיטה parseFloat() המחזירה את הערך המספרי העשרוני (נקודה-צפה) של המחרוזת בשדה.

ניתן להפעיל את השיטה parseInt() על ידי ציון שמה בלבד או על ידי ציון שם האובייקט Global.parseInt().

אם הערך בשדה הוא מחרוזת כמו "2" השיטה parseInt() תחזיר 2. אם הערך בשדה הוא מחרוזת כמו "105.83" השיטה parseFloat() תחזיר 105.83. אם הערך בשדה הוא "juka" אז השיטה parseFloat() תחזיר NaN שפירושו **N**ot **a N**umber. את הערך שמחזירות השיטות parseInt() ו-parseFloat() ניתן לבדוק בעזרת שיטה אחרת, isNaN(), המחזירה true אם הערך אינו מספר ומחזירה false אם הערך מספר. גם שיטה זו שייכת לאובייקט Global.

```
if (isNaN(strFirstName) == false) {
  alert("טעות: הקלדת מספר היכן שצריך להקליד טקסט") ;
}
```

חומר עזר

פירוט כל השיטות והתכונות של אובייקט Global תוכל למצוא בספר **JavaScript סדנת לימוד** בהוצאת **הוד-עמי**, פרק 7 ובנספח א', ובספר **JavaScript המדריך השלם** בהוצאת **הוד-עמי**.

שאלה 7.1

כתוב טופס ובו השדות הבאים:

	סוג שדה	אורך
שם פרטי	text	20
מרכיב משקפיים	Check Box	
מספר עין ימין	text	10
מספר עין שמאל	text	10
לחצן	submit	

א. כבר בעת הצגת הטופס לא ניתן לגשת לשדות "מספר עין ימין" ו- "מספר עין שמאל".

ב. את שדה "שם פרטי" חובה למלא, כך שלא ניתן יהיה לעזוב את השדה בלי שהמשתמש מילא בו לפחות תו אחד.

ג. אם המשתמש סימן את תיבת הסימון, כלומר הוא מרכיב משקפיים, אז יתאפשר לו לגשת לשדות "מספר עין ימין" ו- "מספר עין שמאל".

שינוי תוכן של שדה טקסט

שדה טקסט הוא אובייקט מסוג text ולכן ניתן לגשת לתכונת ה**ערך** שלו (value). כדי לבטא את אובייקט השדה, יש לכתוב את כל ההיררכיה עד אליו, החל מהאובייקט window, דרך אובייקט document, דרך אובייקט הטופס form ועד אובייקט השדה (במקרה זה אובייקט text). למשל, כדי להתייחס לשדה הטקסט שבדוגמה הבאה:

```
<form name="cForm">
  הקלד משפט (בעברית):
  <input type="text" name="sentence" />
</form>
```

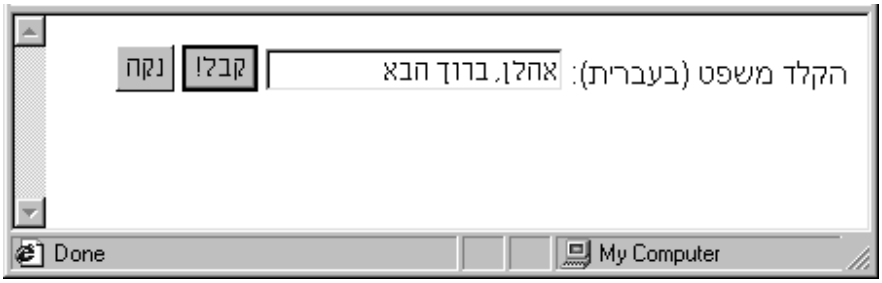
יש לכתוב window.document.cForm.sentence כדי לבטא את מיקומו ההיררכי של האובייקט במודל האובייקטים (DOM). כדי להתייחס אל תכונת הערך ולשנות את הכתוב בשדה, יש לרשום:

```
window.document.cForm.sentence.value = something ;
```

או

```
document.cForm.sentence.value = something ;
```

בדוגמה הבאה נבדק הטקסט המוזן על ידי המשתמש. אם טקסט זה הוא "שלום", יוצב בשדה הטקסט הערך "אהלן, ברוך הבא". אם הטקסט שמוזן על ידי המשתמש שונה, יוצב בשדה הערך "מה אמרת? חזור שנית".



תרשים 7.2: הצגת פלט בשדה טופס.

דוגמה 7-02.html:

```
<html dir="rtl">
 <head>
  <title>7-02</title>
  <script language="javascript">
   function answerPlease()
   {
    if(document.cForm.sentence.value == "שלום") {
     document.cForm.sentence.value = "אהלן, ברוך הבא" ;
    }else{
     document.cForm.sentence.value = "מה אמרת? חזור שנית" ;
    }
   }
  </script>
 </head>
<body>
 <form name="cForm">
  הקלד משפט (בעברית):
  <input type="text" name="sentence" />
  <input type="button" value="קבל!" onclick="answerPlease();"
/>
  <input type="reset" value="נקה" />
 </form>
</body>
</html>
```

שים לב לאובייקטים שנוצרו בטופס זה. התגית <input type="text" /> יצרה אובייקט text, ואילו התגית <input type="button" /> יצרה אובייקט מסוג button. לכל אובייקט יש את השיטות והתכונות שלו.

חומר עזר

חומר לימודי נוסף על מחרוזות ואובייקט String תוכל למצוא בספר **JavaScript סדנת לימוד** בהוצאת **הוד-עמי**, פרק 21 ונספח א'.

שיטות נוספות של אובייקט מחרוזת

השיטה indexOf()

השיטה indexOf() מחזירה את ערך המיקום של המופע הראשון של אותה מחרוזת החיפוש (תו אחד או יותר) במחרוזת בה מחפשים. בעזרת שיטה זו ניתן לחפש מילים או אותיות בודדות בתוך מחרוזת.

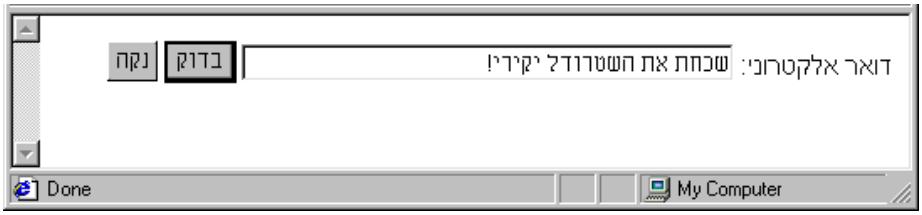
ניצור אובייקט מחרוזת ונפעיל עליו את השיטה ()indexOf, באופן הבא:

```
sText = "Hello" ;
document.write(sText.indexOf("o")) ;
```

התוצאה שנקבל היא 4, מכיון שהאות o (האות הקטנה) שחיפשנו במחרוזת מופיעה בה במקום הרביעי (מחרוזות ב-JavaScript מתחילות את ספירת האותיות באפס). אם הטקסט שחיפשנו אינו מופיע במחרוזת, התוצאה תהיה 1-. שים לב שהמילה Hello מכילה 5 תווים המסומנים מ- 0 עד 4.

הנה דוגמה לאימות הקלדת טקסט של דואר אלקטרוני על ידי שימוש בשיטה ()indexOf למציאת הסימן @. אם השיטה מחזירה 1-, זהו סימן שהמחרוזת אינה מכילה את הסימן, כאשר בכל מקרה אחר הסימן חייב להופיע.

וגם:



תרשים 7.3: בדיקת תקינות שדה דואר אלקטרוני והצגת התשובה בשדה.

דוגמה 7-03.html:

```
<html dir="rtl">
 <head>
  <title>7-03</title>
  <script language="javascript">
   function eMailChecker()
   {
    if (document.seeForm.eMail.value.indexOf("@") == -1) {
      document.seeForm.eMail.value="שכחת את השטרודל יקירי!" ;
     }else{
      document.seeForm.eMail.value = "יפה מאד" ;
    }
   }
  </script>
 </head>
<body>
 <form name="seeForm">
  דואר אלקטרוני:
  <input type="text" name="eMail" size="40" />
  <input type="button" value="בדוק" onclick="eMailChecker();" />
  <input type="reset" value="נקה" />
  </form>
 </body>
</html>
```

מכיון שהשיטה indexOf() מחזירה את המופע הראשון, ייתכן ויש יותר מ- @ אחד בשדה הקלט. את זה הפונקציה לא בודקת.

השיטה lastIndexOf()

שיטה זו מחזירה את מיקום המופע האחרון של מחרוזת החיפוש במחרוזת בה נעשה החיפוש.

```
jNielsen = "Designing Web Usability" ;
eIndx = jNielsen.indexOf("e") ; // equal  1, Designing Web Usability

eLastIndx = jNielsen.lastIndexOf("e") ; // equal 11, Designing Web Usability
```

השיטה split()

שיטה זו מפרקת מחרוזת למערך לפי תו הפרדה. לדוגמה, נתונה מחרוזת:

```
countryUSA = "Alabama, Alaska, California, Florida" ;
```

אפשר לפרק אותה על פי הפסיקים, בצורה הבאה: countryUSA.split(",") . החלוקה בונה מערך והגישה לאיברי המערך מתבצעת כך:

countryUSA.split(",")[0] ייתן את הערך "Alabama",

countryUSA.split(",")[1] ייתן את הערך "Alaska",

countryUSA.split(",")[2] ייתן את הערך "California" וכדומה.

השיטה substring()

שיטה זו גוזרת חלק ממחרוזת. בדוגמה הבאה יוגדר משתנה שהוא אובייקט מחרוזת ותופעל עליו השיטה substring():

```
fullText = "encyclopedia" ;
partText = fullText.substring(4, 8) ;  //  equal  clop, encyclopedia
```

נקבל את תת-המחרוזת "clop", מכיון שתת-המחרוזת תיגזר החל מהתו הרביעי ועד התו השמיני (לא כולל).

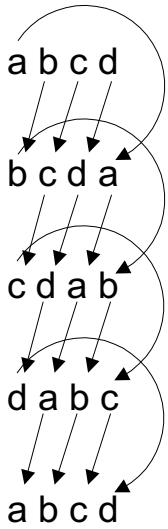
השיטה charAt()

שיטה זו מחזירה תו אחד מהמחרוזת לפי פרמטר שנשלח אליה.

```
bookName = "encyclopedia" ;
pieceName = bookName.substring(3, 4) ;    //  equal  y, encyclopedia
charName = bookName.charAt(3) ;           // equal y, encyclopedia
```

מעניין לראות דוגמה לשימוש בשתי השיטות substring() ו-charAt(), כדי לגלול מחרוזת תווים. הלוגיקה שבגלילת טקסט פשוטה למדי והיא תודגם בעזרת המחרוזת "abcd". מכיון שהטקסט צריך לנוע שמאלה, האלגוריתם יגזור את האות הראשונה ויציב אותה בסוף המחרוזת:

עיצוב ממשק באינטרנט
יצוב ממשק באינטרנטע
צוב ממשק באינטרנטעי
וב ממשק באינטרנטעיצ
ב ממשק באינטרנטעיצו
ממשק באינטרנטעיצוב
ממשק באינטרנטעיצוב
משק באינטרנטעיצוב מ
שק באינטרנטעיצוב ממ
ק באינטרנטעיצוב ממש
באינטרנטעיצוב ממשק
באינטרנטעיצוב ממשק
אינטרנטעיצוב ממשק ב
ינטרנטעיצוב ממשק בא
נטרנטעיצוב ממשק באי
טרנטעיצוב ממשק באינ
רנטעיצוב ממשק באינט
נטעיצוב ממשק באינטר
טעיצוב ממשק באינטרנ
עיצוב ממשק באינטרנט



תרשים 7.4 : שימוש בשיטות substring() ו- charAt().

כלומר, האלגוריתם מרכיב מחרוזת חדשה משתי תת-מחרוזות. המחרוזת הראשונה מורכבת מהאות השנייה (ב-JavaScript זוהי האות במיקום 1, כי התו הראשון נמצא במיקום 0) ועד סוף המחרוזת (את סוף המחרוזת נייצג כ-a.length). למחרוזת זו יש להוסיף את המחרוזת השנייה, המורכבת מהאות הראשונה של המחרוזת המקומית.

השיטה **charAt()** מחזירה תו אחד במחרוזת על פי המיקום הנקבע בין הסוגריים. כלומר, charAt(0) שווה ערך לאות הראשונה במחרוזת.

דוגמה 7-04.html :

```
<html dir="rtl">
 <head>
  <title>7-04</title>
  <script language="javascript">
   function textMover( )
   {
    var sourceText ;
    var newText ;
    sourceText = document.moveForm.text2move.value ;
    newText = sourceText.substring(1,sourceText.length) ;
    newText = newText + sourceText.charAt(0) ;
    document.moveForm.text2move.value = newText ;
   }
```

```
  </script>
 </head>
 <body>
  <form name="moveForm">
   הקלד טקסט והתחל לגלגל...
   <br />
   <input type="text" name="text2move" value="" />
   <input type="button" value="גלגל" onclick="textMover();" />
   <input type="reset" value="נקה" />
  </form>
 </body>
</html>
```

בדוגמה זו, הלחצן בטופס קורא לפונקציה ()textMover אשר מרכיבה מחרוזת חדשה על ידי שרשור שתי מחרוזות, וקובעת את המחרוזת החדשה כערך של שדה הטקסט בטופס. הפונקציה שבדוגמה **7-04.html** פועלת רק פעם אחת, ולכן יש ללחוץ על הלחצן **גלגל** עבור כל תנועה.

חומר עזר

פירוט כל השיטות והתכונות של אובייקט String תוכל למצוא בספר **JavaScript סדנת לימוד** בהוצאת **הוד-עמי**, נספח א', ובספר **JavaScript המדריך השלם** בהוצאת **הוד-עמי**.

שאלה 7.2

כתוב טופס ובו השדות הבאים:

	סוג שדה	אורך
שם פרטי	text	15
שם משפחה	text	22
מיקוד	text	5
כתובת אתר אינטרנט	text	35
לחצן	submit	
לחצן	reset	

א. חובה למלא שם פרטי.

ב. חובה למלא שם משפחה.

ג. בדוק ששדה המיקוד מכיל 5 ספרות.

ד. כתובת אתר האינטרנט היא מסוג: www.hod-ami.co.il, www.microsoft.com, java.sun.com וכדומה. בדוק שכתובת אתר האינטרנט חוקית:

1. לפחות 2 נקודות (.).
2. לפחות 2 אותיות בכל קבוצה.
3. אינה מכילה את התווים הבאים: !@$^*)(~}]{[;<>

השיטה **setInterval()** מאפשרת הפעלה של פונקציה במרווח זמן בפעולה מחזורית ואותה הכרת בפרק הקודם. קובץ **7-05.html** הוא שיפור של קובץ **7-04.html**, בכך שהוא מפעיל שוב ושוב את אותה הפעולה הלוקחת את התו האחרון במחרוזת ומדביקה אותו לסופה. ניתן גם להוסיף לחצן לעצירת הפעולה בעזרת השיטה clearInterval(), במידה והפונקציה setInterval() מופעלת במשפט הצבה לתוך משתנה.

דוגמה 7-05.html:

```
<html dir="rtl">
 <head>
  <title>7-05</title>
  <script language="javascript">
   var x ;
   function initTextMover()
   {
    if (document.moveForm.text2move.value.length != 0) {
     x = setInterval('textMover()',600) ;
    }else{
     window.alert("צריך להיות ערך בשדה") ;
     document.moveForm.text2move.focus() ;
    }
   }

   function textMover( )
   {
    var sourceText ;
    var newText ;
    sourceText = document.moveForm.text2move.value ;
    newText = sourceText.substring(1,sourceText.length) ;
    newText = newText + sourceText.charAt(0) ;
```

```
    document.moveForm.text2move.value = newText ;
   }
  </script>
 </head>
 <body>
  <form name="moveForm">
   הקלד טקסט והתחל לגלגל...
   <br />
   <input type="text" name="text2move" value="" />
   <input type="button" value="גלגל"
     onclick="initTextMover();" />
   <input type="reset" value="נקה" />
   <input type="button" value="הפסק לגלגל"
     onclick="clearInterval(x);" />
  </form>
 </body>
</html>
```

הפונקציה initTextMover() בודקת מה אורך המחרוזת בשדה הטופס. אם האורך שונה מאפס, סימן שיש תו אחד או יותר בשדה, ובמקרה זה יש להפעיל את הפונקציה setInterval(). אם השדה ריק (אורכו שווה לאפס), מוצגת למשתמש הודעה באמצעות השיטה window.alert() והמיקוד עובר לשדה הטופס:

```
window.alert("צריך להיות ערך בשדה") ;
document.moveForm.text2move.focus() ;
```

שים לב שהמשתנה x הוגדר מחוץ לפונקציה כדי שיהיה משתנה גלובלי וכל תסריט בדף יוכל להתייחס אליו.

השיטות toUpperCase() ו- toLowerCase()

השיטות **toUpperCase()** ו-**toLowerCase()** משמשות להסבת המחרוזת מאותיות רישיות (גדולות) לאותיות רגילות (קטנות) ולהיפך, לדוגמה:

```
USA = "aMeRica" ;
window.alert(USA.toLowerCase() + " " + USA.toUpperCase()) ;
```

תרשים 7.5: שיטות שונות של אובייקט String.

שאלה 7.3

כתוב טופס עם שדה אחד (רק שדה אחד, ללא לחצן כלשהו) לקליטת מספר. על המספר להיות בין 10 ועד 15 (כולל).

א. בדיקת השדה תיעשה עם עזיבתו על ידי לחיצת מקש Tab.

ב. אם הערך בשדה אינו בטווח תופיע ההודעה:

```
Error: must be between 10 and 15
```

חומר עזר

פירוט כל השיטות והתכונות של אובייקט Text תוכל למצוא בספר **JavaScript סדנת לימוד** בהוצאת **הוד-עמי**, פרק 24, 25, 26 ובנספח ב', ובספר **JavaScript המדריך השלם** בהוצאת **הוד-עמי**.

טיפול בנתוני טופס (המשך)

רשימה נגללת select

לרשימה נגללת יש לגשת באופן שונה מעט מאשר אל שדות טקסט. התגית <select> יוצרת אובייקט בשם Select. כדי לברר איזה ערך נבחר ברשימה, יש לגשת תחילה אל האובייקט Select, בדומה לגישה אל אובייקט Text.

```
<form name="Form4U">
   <select name="dropList">
    <option>
     HTML 4 למפתחי אתרים באינטרנט
    </option>
    <option>
     JavaScript סדנת לימוד
    </option>
    <option>
     JavaScript+CSS+DOM למפתחי אתרים באינטרנט
    </option>
    <option>
     ASP 3 סדנת לימוד
    </option>
   </select>
   <input type="button" onclick="showSelect();" value="פרטים" />
</form>
```

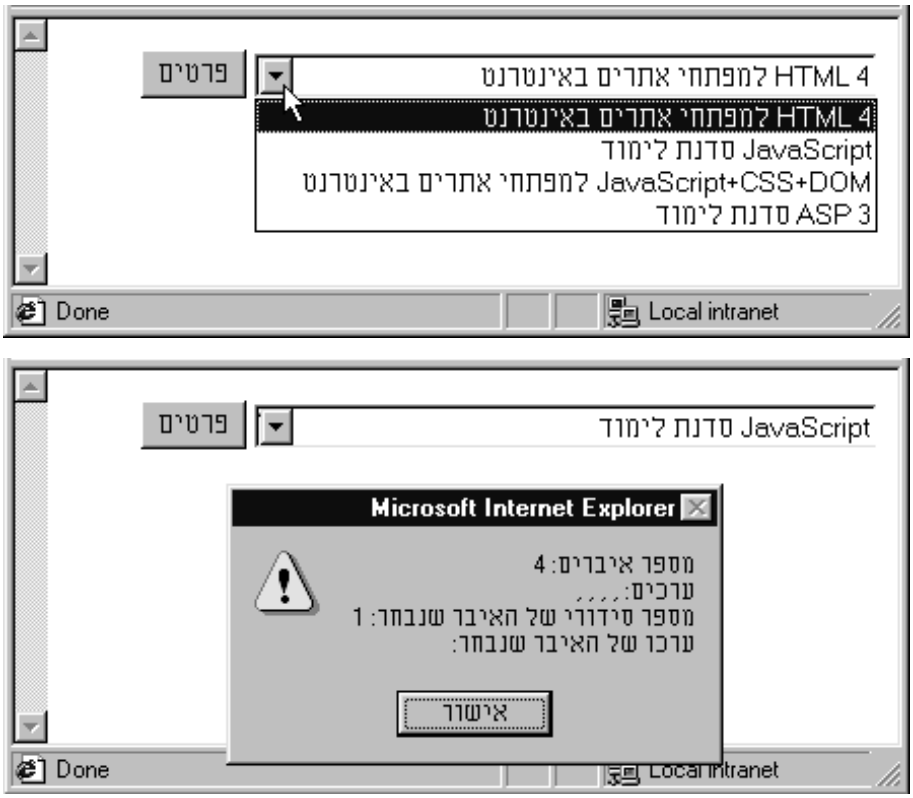
אל השדה מסוג select, הנקרא dropList אשר בטופס Form4U, ניתן לגשת כ- document.Form4U.dropList. אולם, כדי לברר את הערך הנבחר, יש להשתמש בשתי תכונות. התכונה הראשונה היא selectedIndex, המציינת איזו מהאפשרויות (המסודרות **כמערך**) נבחרה.

כלומר, אם נבחר באפשרות (האיבר) **HTML 4 למפתחי אתרים באינטרנט** ונציג את document.Form4U.dropList.selectedIndex נקבל 0, כיון שהאיבר הראשון במערך מסומן כמספר 0. הדוגמה הבאה ממחישה את השימוש בתכונה selectedIndex.

דוגמה 7-06.html:

```
<html dir="rtl">
 <head>
  <title>7-06</title>
  <script language="javascript">
   function showSelect()
   {
     noOfItems = document.Form4U.dropList.length ;
     Items = "" ;
```

```
      for (var i = 0 ; i < noOfItems ; i++ ) {
       Items = Items + document.Form4U.dropList.options[i].value ;
       Items = Items + ", " ;
      }
      noOptItem = document.Form4U.dropList.selectedIndex ;
      valueOfselItem = document.Form4U.dropList.options[noOptItem].value ;

   Msg = "" ;
   Msg = Msg + "מספר איברים: " + noOfItems + "\n" ;
   Msg = Msg + "ערכים: " + Items + "\n" ;
   Msg = Msg + "מספר סידורי של האיבר שנבחר: " + noOptItem + "\n" ;
   Msg = Msg + "ערכו של האיבר שנבחר: " + valueOfselItem ;
      window.alert(Msg) ;
    }
   </script>
  </head>
  <body>
   <form name="Form4U">
    <select name="dropList">
     <option>
      HTML 4 למפתחי אתרים באינטרנט
     </option>
     <option>
      JavaScript סדנת לימוד
     </option>
     <option>
      JavaScript+CSS+DOM למפתחי אתרים באינטרנט
     </option>
     <option>
      ASP 3 סדנת לימוד
     </option>
    </select>
    <input type="button" onclick="showSelect();" value="פרטים" />
   </form>
  </body>
 </html>
```



תרשים 7.6: בעזרת גישה ל-DOM ניתן לדעת מה המשתמש בחר בשדה מסוג Select.

כפי שניתן לראות בתמונות המסך, selectedIndex מחזירה את המספר הסידורי של הבחירה במערך האפשרויות המוצג. המספר 1 מציין שנבחר האיבר... השני במערך.

כדי לקבל את הערך של האפשרות (האיבר) שנבחרה ולא את המספר הסידורי שלה, יש לפנות אל הרשימה כמערך:

```
document.Form4U.dropList.options[the number you want].value
```

המספר שיצוין בין הסוגריים המרובעים יפנה אל האפשרות המתאימה במערך select.
אם רוצים לקבל את הערך של האפשרות שנבחרה על ידי המשתמש, יש לכתוב:

```
document.Form4U.dropList.options[document.Form4U.dropList.selectedIndex].value
```

המשפט document.Form4U.dropList.selectedIndex ייתן את המספר הסידורי של האפשרות הרצויה.

הבעיה בדוגמה **7-06.html** היא שלא התקבלו ערכים מאובייקט select. ניתן לדעת כמה איברים יש באובייקט select וגם ניתן לדעת את מספרו הסידורי של האיבר הנבחר, אבל מה הערך של האיברים באובייקט select? לשאלה זו קובץ **7-06.html** אינו נותן מענה.

בהגדרת האובייקט select אין ערך לאפשרויות השונות, מכיון שמאפיין value של תגית <option> לא הוגדר.

שאלה ותשובה!

האם ניתן לקבל את ערך הבחירה גם ללא המאפיין value בתגית <option>?

כן, ניתן בצורה הבאה:

```
noOptItem = document.Form4U.dropList.selectedIndex ;
```

ואחר כך לעשות שימוש בערך:

```
document.Form4U.dropList.options[noOptItem].innerText
```

פירוט על innerText תוכל למצוא בספר **JavaScript+CSS+DOM למפתחי אתרים באינטרנט** בהוצאת **הוד-עמי**.

לכן, יש להגדיר את האובייקט select ולהציב ערכים במאפיין value של התגית <option>:

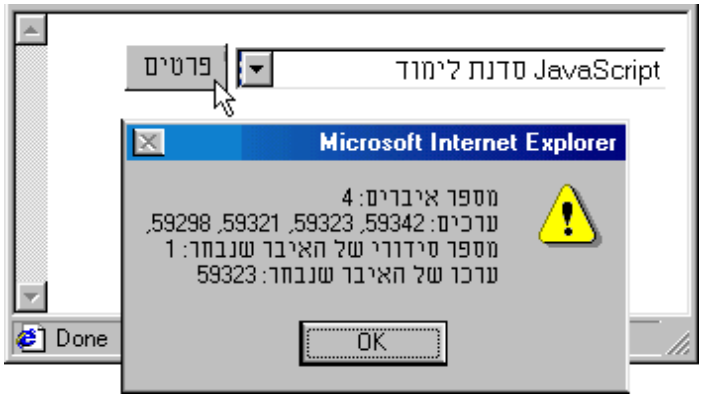
```
<select name="dropList">
 <option value="59342">
  HTML 4 למפתחי אתרים באינטרנט
 </option>
 <option value="59323">
  JavaScript סדנת לימוד
 </option>
 <option value="59321">
  JavaScript+CSS+DOM למפתחי אתרים באינטרנט
 </option>
 <option value="59298">
  ASP 3 סדנת לימוד
 </option>
</select>
```

להלן הקוד המלא של קובץ **7-07.html** המציג את כל הערכים של שדה select:

```
<html dir="rtl">
 <head>
  <title>7-07</title>
  <script language="javascript">
   function showSelect()
   {
     noOfItems = document.Form4U.dropList.length ;
     Items = "" ;
     for (var i = 0 ; i < noOfItems ; i++ ) {
      Items = Items + document.Form4U.dropList.options[i].value ;
      Items = Items + ", " ;
     }
     noOptItem = document.Form4U.dropList.selectedIndex ;
     valueOfselItem = document.Form4U.dropList.options[noOptItem].value ;

     Msg = "" ;
     Msg = Msg + "מספר איברים: " + noOfItems + "\n" ;
     Msg = Msg + "ערכים: " + Items + "\n" ;
     Msg = Msg + "מספר סידורי של האיבר שנבחר: " + noOptItem + "\n" ;
     Msg = Msg + "ערכו של האיבר שנבחר: " + valueOfselItem ;
     window.alert(Msg) ;
   }
  </script>
 </head>
 <body>
  <form name="Form4U">
   <select name="dropList">
    <option value="59342">
     למפתחי אתרים באינטרנט HTML 4
    </option>
    <option value="59323">
     סדנת לימוד JavaScript
    </option>
    <option value="59321">
     למפתחי אתרים באינטרנט JavaScript+CSS+DOM
    </option>
    <option value="59298">
     סדנת לימוד ASP 3
    </option>
   </select>
   <input type="button" onclick="showSelect();" value="פרטים" />
```

```
  </form>
 </body>
</html>
```

לפניך תמונות המסך של הדוגמה. שים לב, הפעם נקבע ערך שדה הטקסט לפי ערך הבחירה ולא לפי מספרה הסידורי.



תרשים 7.7 : פנייה לערך ברשימה בשדה מסוג Select.

חומר עזר

פירוט כל השיטות והתכונות של אובייקט Select תוכל למצוא בספר **JavaScript סדנת לימוד** בהוצאת **הוד-עמי**, פרק 26 ובנספח ב', ובספר **JavaScript המדריך השלם** בהוצאת **הוד-עמי**.

שאלה 7.4

בנה טופס ובו שני שדות:

- שדה אחד מסוג select המכיל את רשימת מדינות העולם (תוכל להיעזר בקובץ Country.txt הנמצא בין הקבצים של פרק זה, אותם ניתן להוריד מאתר האינטרנט של הספר בכתובת www.hod-ami.co.il/59355.html).
- שדה שני מסוג text שאינו ניתן לגישה, בו תציג את מספר המדינה שנבחרה בשדה הראשון.

א. מהו מספרה של ישראל (Israel)? _____

ב. מהו מספרה של קולומביה (Colombia)? _____

ג. מהו מספרה של טהיטי (Tahiti)? _____

שאלה 7.5

בנה טופס ובו שני שדות:

- שדה ראשון מסוג select המכיל את רשימת הערים בגוש דן (היעזר בקובץ IsraelCityCenter.txt הנמצא בין הקבצים של פרק זה, אותם ניתן להוריד מאתר האינטרנט של הספר בכתובת www.hod-ami.co.il/59355.html).
- שדה שני מסוג text שאינו ניתן לגישה, בו תציג את מספר העיר שנבחרה בשדה הראשון.

א. מהו מספרה של "ברקן"? _____

ב. מהו מספרה של "עינת"? _____

ג. מהו מספרה של "חולון"? _____

לחצני רדיו - Radio buttons

הפנייה בתוכנית אל לחצני רדיו כדי ללמוד מי מהם נלחץ היא פעולה מסובכת יותר, כי יש לבדוק כל אחד מהם. לחצן שנלחץ מקבל ערך true בתכונה checked.

לחצני הרדיו, בדומה לפריטים ברשימת הגלילה, מסודרים כאיברים של מערך. יש להריץ לולאה על מערך הלחצנים הזה, מ- 0 עד לאיבר האחרון של המערך. בדוגמה הבאה, יש לולאה הפועלת על איברי המערך louis המכילים את הלחצנים ובודקת את ערך התכונה checked של כל לחצן. רק בלחצן הנבחר, ערך התכונה checked יהיה true.

יצירת מערך לחצנים בשם louis:

```
<form name="myForm" method="post"
action="mailto:mooki@smoopi.co.il">
  לואי ארמסטרונג היה:
  <br />
  <input type="radio" name="louis" value="niel" checked="checked" />
```

```
   "אסטרונאוט שאמר "זהו צעד קטן לאדם, אך זינוק אדיר לאנושות
  <br />
  <input type="radio" name="louis" value="Correct" />
   זמר, נגן חצוצרה ומלחין שנולד וגר בניו אורלינס
  <br />
  <input type="radio" name="louis" value="king" />
   גדול המנהיגים השחורים בארה"ב, לוחם למען זכויות האדם והשלום
  <br />
  <input type="radio" name="louis" value="reporter" />
   סופר אמריקאי שבזכות ספרו ה"זקן והים" זכה בפרס נובל
</form>
```

מכיון שהלחצנים מהווים מערך, הלחצן הראשון מסומן 0. לכן, אם בבדיקה יתקבל הערך 0, נדע שהמשתמש בחר ב- **אסטרונאוט..** וכן הלאה. מספר האיברים במערך נמצא בתכונה length של אובייקט louis ונציב אותו במשתנה noOfItems, כך:

```
noOfItems = document.myForm.louis.length ;
```

כדי לדעת איזו אפשרות נבחרה, יש צורך בלולאה שתעבור על ערכי האוסף ותבדוק את התכונה checked של כל איבר ואיבר. שים לב שבטופס נבחרה האפשרות הראשונה. המשמעות היא שהטופס נשלח עם אפשרות שנבחרה וזו יכולה להיות אפשרות שבחר הגולש או אפשרות שנבחרה עבורו, במסגרת הטופס, והוא לא שינה את הבחירה:

```
for(i = 0 ;  i < noOfItems ;  i++)
{
 if(document.myForm.louis[i].checked) {
  itemSeqNo = i ;
  itemValue = document.myForm.louis[i].value ;
 }
}
```

דוגמה 7-08.html:

```
<html dir="rtl">
 <head>
  <title>7-08</title>
  <script language="javascript">
   function showRadio()
   {
     noOfItems = document.myForm.louis.length ;

     Items = "" ;
     for (var i = 0 ; i < noOfItems ; i++ ) {
      Items = Items + document.myForm.louis[i].value ;
      Items = Items + ", " ;
     }

     for(i = 0 ;  i < noOfItems ;  i++)
```

```
      {
       if(document.myForm.louis[i].checked) {
        itemSeqNo = i ;
        itemValue = document.myForm.louis[i].value ;
       }
      }

      Msg = "" ;
      Msg = Msg + "מספר איברים: " + noOfItems + "\n" ;
      Msg = Msg + "ערכים: " + Items + "\n" ;
      Msg = Msg + "מספר סידורי של האיבר שנבחר: " + itemSeqNo + "\n" ;
      Msg = Msg + "ערכו של האיבר שנבחר: " + itemValue ;
      window.alert(Msg) ;

      if (itemSeqNo == 1) {
       window.alert("תשובה נכונה") ;
      }else{
       window.alert("לא, אבל קרוב") ;
      }
    }
  </script>
 </head>
<body>
 <form name="myForm" onsubmit="showRadio();">
  לואי ארמסטרונג היה:
  <br />
  <input type="radio" name="louis" value="niel" checked="checked" />
   אסטרונאוט שאמר "זהו צעד קטן לאדם, אך זינוק אדיר לאנושות"
  <br />
  <input type="radio" name="louis" value="Correct" />
   זמר, נגן חצוצרה ומלחין שנולד וגר בניו אורלינס
  <br />
  <input type="radio" name="louis" value="king" />
   גדול המנהיגים השחורים בארה"ב, לוחם למען זכויות האדם והשלום
  <br />
  <input type="radio" name="louis" value="reporter" />
   סופר אמריקאי שבזכות ספרו ה"זקן והים" זכה בפרס נובל
  <br />
  <input type="submit" value="שלח" />
 </form>
</body>
</html>
```

תרשים 7.8: דוגמה לשימוש בטופס ליצירת שאלון אמריקאי.

חומר עזר

פירוט כל השיטות והתכונות של אובייקט Radio תוכל למצוא בספר **JavaScript סדנת לימוד** בהוצאת **הוד-עמי**, פרק 26 ובנספח ב', ובספר **JavaScript המדריך השלם** בהוצאת **הוד-עמי**.

שאלה 7.6

בנה טופס ובו שני שדות:

- שדה ראשון מסוג select המכיל רשימת מזון עם אפשרות בחירה מרובה. המאפיין value של התגית <option> מציין את מספר הקלוריות ל-100 גרם של מזון זה (היעזר בקובץ cal.txt הנמצא בין הקבצים של פרק זה, אותם ניתן להוריד מאתר האינטרנט של הספר בכתובת www.hod-ami.co.il/59355.html).
- שדה שני מסוג text שאינו ניתן לגישה, בו תציג את <u>מספר הקלוריות הממוצע</u> של המזונות שנבחרו בשדה הראשון.

א. מהו ממוצע הקלוריות של שומשום, משמשים מיובשים ואגוזי אילסר? _____

ב. מהו ממוצע הקלוריות של שקדים וזרעוני חמניות? _____

ג. מהו ממוצע הקלוריות של תאנים מיובשים ומשמשים מיובשים? _____

שאלה 7.7

בקובץ WhoSaid.html (הנמצא בין הקבצים של פרק זה, אותם ניתן להוריד מאתר האינטרנט של הספר בכתובת www.hod-ami.co.il/59355.html) תמצא שתי שאלות. התשובה לשאלה הראשונה היא "ביל גייטס" והתשובה לשאלה השנייה היא "סנטור בוב דול". דאג שלחיצה על הלחצן OK תבדוק את תשובת המשתמש לשתי שאלות אלו ותתן לו ציון בהתאם.

חומר עזר

פירוט כל השיטות והתכונות של אובייקט Reset, Submit, CheckBox ו-Form תוכל למצוא בספר **JavaScript סדנת לימוד** בהוצאת **הוד-עמי**, ובספר **JavaScript המדריך השלם** בהוצאת **הוד-עמי**.

פרויקט סיינפלד

למעשה, הרעיון הראשוני והמרכזי סביב פיתוח JavaScript היה עבודה עם טפסים. טופס ההרשמה למועדון הלקוחות של סיינפלד מכיל שדות מסוגים שונים, שבחלקם צריך לערוך בדיקות תקינות. כלומר, לוודא שהוכנס ערך שהוא נכון (לפחות מבחינת התחביר) לשדה. למשל, בשדה דואר אלקטרוני לא נוכל לבדוק האם הכתובת האלקטרונית אכן קיימת, אבל נוכל לבדוק שהיא מכילה את הסימן @, כי בלי סימן זה, זו אינה כתובת אלקטרונית.

מלבד בדיקת התקינות לשדה, יש להחליט עוד שתי החלטות:

1. מתי נערכת הבדיקה? בעת לחיצה על לחצן **שלח טופס** או מייד לאחר הקלדת השדה?
2. מה עושים במידה והבדיקה נכשלה, כלומר ערך השדה שגוי.

לצורך הדוגמה, הוחלט ש:

1. הבדיקה תיערך בעת לחיצה על לחצן **שלח טופס**.
2. כל ההערות יוצגו למשתמש בתיבה ()alert.

להלן בדיקות התקינות שיש לערוך לטופס שבדף Club.html:

שדה	מהות הבדיקה
שם	לא ריק
דואר אלקטרוני	תחביר
מתי לראשונה ראית את הסדרה	בחירה ולא <<בחר שנה>>
ההורים הכי מצחיקים	בחירה

נהוג לסמן בטופס את השדות שחובה למלא. נעשה זאת על ידי ציון כוכבית אדומה ליד שם השדה בתוספת הערה. לדוגמה, כך נוספה כוכבית אדומה לשדה שם:

```
<tr>
  <td>שם
    <sup><font color="red">*</font></sup>
  </td>
  <td>
    <input name="name" type="text" size="25" />
  </td>
</tr>
```

לכל סוג בדיקה יש לבנות פונקציית בדיקה:

שדה	הפונקציה שבודקת
שם	checkEmpty()
דואר אלקטרוני	checkEmail()
מתי לראשונה ראית את הסדרה	checkYear()
ההורים הכי מצחיקים	checkParents()

לפני שמתחילים, כדאי להיזכר בשמות השדות בטופס שבדף Club.html.

האובייקט	שם האובייקט
טופס	seinfeldForm
שדה: שם	name
שדה: דואר אלקטרוני	eMail
שדה: מתי לראשונה	firstYear
שדה: ההורים הכי מצחיקים	parents

מכיון שהשדות ייבחנו בעת לחיצה על לחצן **שלח טופס**, יש לאסוף את כל הודעות השגיאה במהלך הבדיקות למשתנה גלובלי, כדי שכל הפונקציות יכירו אותו. נקרא למשתנה הגלובלי הזה בשם formErrors.

להלן הקוד של הפונקציה checkEmpty():

```
function checkEmpty()  {
  var name ;
  name = window.document.seinfeldForm.name.value ;
  if (name.length > 0) {
    return true ;
   }else{
    formErrors += 'שם ריק' + '\n' ;
    return false ;
   }
}
```

שימוש בתכונה length כדי לבחון את אורך השדה. הפונקציה מחזירה false או true באמצעות משפט return בהתאם לתוצאות הבדיקה (נושא החזרת ערך מפונקציה הוסבר בפרק 5).

להלן קוד המקור של הפונקציה checkEmail():

```
function checkEmail()  {
  var eMail ;
  eMail = window.document.seinfeldForm.eMail.value ;
  if (eMail.length > 0)  {
    if ((eMail.indexOf('@')>0) && (eMail.indexOf('.')>0)) {
      return true ;
     }else{
      formErrors += 'דואר אלקטרוני לא כתוב נכון' + '\n' ;
      return false ;
    }
   }else{
    formErrors += 'דואר אלקטרוני ריק' + '\n' ;
    return false ;
   }
}
```

בפונקציה זו נעשות שתי בדיקות. בדיקה אחת של שדה ריק ובדיקה שנייה להימצאות התווים @ ו- . (נקודה) - שניהם חייבים להימצא ולכן התנאי הוא && שפירושו and.

להלן הקוד של הפונקציה checkYear():

```
function checkYear()  {
 var year2Select ;
 year2Select = window.document.seinfeldForm.firstYear.selectedIndex ;
 if (year2Select > 0) {
   return true ;
  }else{
   formErrors += 'לא בחרת שנה' + '\n' ;
   return false ;
 }
}
```

שדה firstYear בטופס הוא שדה מסוג select וזאת עליך לדעת:

פנייה לאובייקט select	document.seinfeldForm.firstYear
מספר האיברים באובייקט	document.seinfeldForm.firstYear.length
מערך ערכי האיברים שבאובייקט	document.seinfeldForm.firstYear.options
פנייה לערך הראשון	document.seinfeldForm.firstYear.options[0].value
מספרו הסידורי של הערך שנבחר	document.seinfeldForm.firstYear.selectedIndex

להלן הפונקציה checkParents():

```
function checkParents()  {
  var parents0 ;
  var parents1 ;
  parents0 = window.document.seinfeldForm.parents[0].checked ;
  parents1 = window.document.seinfeldForm.parents[1].checked ;
  if (parents0 || parents1) {
    return true ;
   }else{
    formErrors += 'לא בחרת הורים מצחיקים' + '\n' ;
    return false ;
  }
}
```

שדה מסוג radio הוא למעשה אוסף של שדות באותו שם. הטופס שבדף Club.html מכיל שני שדות מסוג <input type="radio" /> באותו שם - parents.

כדי שהבדיקה תצליח, צריך להיבחר שדה אחד לפחות. כלומר, הערך של התכונה checked צריך להיות true. לכן, התנאי הוא || שפירושו OR.

ועכשיו צריך לארגן את הבדיקות לרצף אחד. על זה אחראית הפונקציה checkForm():

```
function checkForm()  {
 formErrors = '' ;
 checkEmpty() ;
 checkEmail() ;
 checkYear() ;
 checkParents() ;
 if (formErrors.length == 0)  {
   return true ;
  }else{
   alert(formErrors) ;
   window.document.seinfeldForm.name.focus() ;
   return false ;
 }
}
```

מייד עם הפעלת הפונקציה checkForm() מאפסים את הערך של formErrors ומפעילים את פונקציות הבדיקה זו אחר זו.

הערך, אם ישנו, במשתנה formErrors הוא אוסף השגיאות. במידה ויש ערך בשדה זה הוא מוצג, והסמן מוצב בשדה **שם** (name) בטופס בעזרת השיטה focus().

ואיך מופעלת הפונקציה checkForm()?

```
<form name="seinfeldForm" onsubmit="return checkForm();">
```

בתגית <form> הוסף את האירוע onsubmit שמופעל בעת לחיצה על לחצן **שלח טופס**. שים לב למילה return שנוספה. שיטת כתיבה זו מבטיחה את הקשר בין התשובה שחזרה (true או false) לבין הערך של המאפיין action (שעדיין לא נכתב). במאפיין action יש לרשום מה לעשות עם הטופס (לשלוח אותו לשרת, כמובן). כאמור, פעילות זו תתבצע רק כאשר תוחזר תשובה true מהאירוע onsubmit.

וכעת לדף Trivia.html, שגם הוא טופס. כאשר המשתמש לוחץ על לחצן **שלח טופס**, תופיע תיבת דו-שיח עם התוצאה.

הדף Trivia.html מורכב מ- 4 שאלות. כל שאלה מכילה 3 תשובות. לכל שאלה אפשר לבחור רק תשובה אחת נכונה, ולכן נעשה שימוש בשדה radio ולא ב-checkbox. לכל קבוצה של 3 שאלות ניתן אותו שם: answer01, answer02, answer03 ו- answer04.

מכיון שיש 3 שדות שקוראים להם answer01, למעשה יש מערך:

```
document.trivia.answer01[0]
document.trivia.answer01[1]
document.trivia.answer01[2]
```

באופן דומה לכל קבוצה אחרת:

```
document.trivia.answer02[0]
document.trivia.answer02[1]
document.trivia.answer02[2]
```

התשובות הנכונות הן:

1 ב': בניה "נדבק" לג'רי ויחד הם הלכו למסעדה. אח"כ בניה טען שהזמין מרק וזה לא נחשב ארוחה...

2 ג': טים התגייר בגלל הבדיחות (על יהודים).

3 ב': קרמר הגה את הרעיון שכל אחד יוכל לאפות את הפיצה שלו. לשם כך הוא התחבר לפפי - המסעדן שלא שוטף ידים לאחר שהיה בשירותים. הוא גם זכור לטוב מהפרק בו השאיר כתם על הספה של ג'רי...

להיראות עסוק זה רעיון בלעדי של ג'ורג שחיפש כל דרך רק כדי שהמנהלים שלו לא יטילו עליו משימות.

החזיה לגברים, עוד רעיון משוגע של קרמר כשהפעם השותף הוא אבא של ג'ורג.

העשרה: לקרמר היו רעיונות נוספים, כמו למשל: ספר בנושא קפה היכול לשמש גם כשולחן קפה, שינת דה וינצ'י בה ישנים 20 דקות כל 3 שעות, ללבוש בגד שחומם בתנור פיצה, מכונית עם פריסקופ לניווט בעיר ועוד ועוד.

4 א': אמא של קרמר (ששמה הוא babs), זאת שניומן התעסק איתה, היתה מנקה.

להלן הפונקציה לבדיקת התשובות בדף Trivia.html:

```
<script language="JavaScript">
   function checkPlease() {
     var points ;
     points = 0 ;
     if (document.trivia.answer01[1].checked) points++ ;
     if (document.trivia.answer02[2].checked) points++ ;
     if (document.trivia.answer03[1].checked) points++ ;
     if (document.trivia.answer04[0].checked) points++ ;
     points = parseInt(points*100/4) ;
     alert ("הציון שלך: " + points) ;
     return false;
   }
</script>
```

ואיך מופעלת הפונקציה checkPlease()?

```
<form name="trivia" onsubmit="return checkPlease();">
```

שים לב למילה return. אם הפונקציה תחזיר true אז יתבצע הכתוב תחת מאפיין action בתגית <form>, ואם הפונקציה תחזיר false אז לא יתבצע הכתוב במאפיין action. אפשר גם בלי return, אבל צריך לדעת מה אתה רוצה שיתבצע לאחר לחיצת לחצן **שלח טופס**. אם הטופס מיועד לעבודה במחשב הלקוח ללא קשר כלשהו לשרת, אז אין צורך ב-return. אבל, אם יש צורך לשלוח את פרטי הטופס, יהיה צורך גם להוסיף את מאפיין action ולדאוג שהפונקציה תדע להחזיר true.

העשרה

המשך לנושא קיבוע הלחצנים, קובץ menuADV.html מהפרק הקודם.

הבעיה עדיין לא נפתרה. הלחצן לא חוזר למצבו. הסיבה היא שבעת לחיצה על הלחצן התמונה שלו היא **on**Button, בעוד אתה רוצה להחזיר אותו למצב **off**Button.

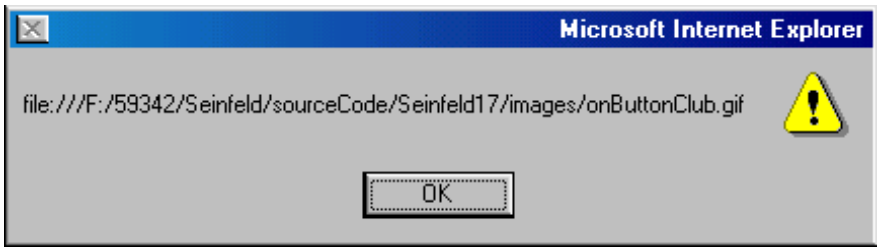
לכן, יהיה עליך לשנות את שם התמונה במשפט:

```
iClickThisButtonSRC = buttonObj.src ;
```

אם תרשום את משפט alert() הבא:

```
alert(iClickThisButtonSRC) ;
```

תוכל לראות את שם התמונה (כולל הנתיב). במחשב שלי זה נראה כך:



תרשים 7.9

בהנחה שהתמונות נמצאות באותו מקום, נוכל לאתר את רצף התווים onButton ולהחליף אותו בתווים offButton, באופן הבא:

```
lenSRC = iClickThisButtonSRC.length ;
foundOnButton = iClickThisButtonSRC.indexOf('onButton') ;
if (foundOnButton != -1) {
 newSRC = iClickThisButtonSRC.substring(0, foundOnButton)
          ↳ + 'offButton' ;
 for (i = foundOnButton+8 ;  i <= lenSRC ;  i++)  {
  newSRC += iClickThisButtonSRC.charAt(i) ;
 }
 iClickThisButtonSRC = newSRC ;
}
```

פרק 8

אובייקט התאריך

לעיתים יש צורך במידע על זמן, כמו שעה או תאריך, לצרכי תצוגה או חישובים. JavaScript מכילה אובייקט מיוחד המספק את כל המידע הקשור בזמן, זהו אובייקט **תאריך** (Date object). את אובייקט התאריך (Date) ניתן לייצר בדומה לאובייקט התמונה (Image) על ידי שימוש במילה השמורה new.

```
Today = new Date() ;
```

אם תבקש להציג את האובייקט תקבל מחרוזת המפרטת את הזמן המדויק כולל שניות, דקות, שעות ותאריך. לדוגמה:

```
Mon Feb 12 21:14:37 UTC+0200 2001
```

הדוגמה הבאה תציג בראש מסמך את מחרוזת התאריך.

דוגמה 8-01.html:

```
<html>
 <head>
  <title>8-01</title>
  <script language="javascript">
   Today = new Date() ;
   document.write(Today) ;
  </script>
 </head>
<body>
</body>
</html>
```

כך תיראה התוצאה בדפדפן:

תרשים 8.1: זוהי מחרוזת התאריך. מה השעה בבקשה?

ברגע שנוצר אובייקט מסוג Date, ניתן להשתמש בשיטות להצגת הגורמים השונים המרכיבים את האובייקט. אם תרצה לברר את השנה הנוכחית, תיעזר בשיטה **getYear()** באופן הבא:

```
myDay = new Date() ;
document.write(myDay.getYear()) ;
```

שיטות נוספות של האובייקט **Date()**:

השיטה	טווח לרבים
getMonth()	0..11
getDate()	1..31
getHours()	0..23
getMinutes()	0..59
getSeconds()	0..59

הערה

בשיטה **getMonth()** ספירת החודשים מתחילה ב- 0.

חומר עזר

פירוט כל השיטות והתכונות של אובייקט Date תוכל למצוא בספר **JavaScript סדנת לימוד** בהוצאת **הוד-עמי**, פרק 21 ונספח א', ובספר **JavaScript המדריך השלם** בהוצאת **הוד-עמי**.

שעון

כדי ליצור שעון יש לבחור את דרך הצגתו. כלומר, אם תבחר להציג את השעה בעזרת השיטה ()document.write, לא תוכל לעדכן את השעון אלא להציגו באופן חד-פעמי, מכיון שהשיטה ()write רק מחוללת מסמך HTML ואינה מסוגלת לעדכן אותו. כדי להציג את השעה וגם לעדכן אותה בצורה דינמית, תוכל להציב אותה כערך של שדה טקסט. אז, ניתן לעדכן את השדה בצורה דינמית על ידי קביעת ערך התכונה value שלו. לחילופין, תוכל להציג את השעה בשורת המצב על ידי קביעת ערך לתכונה window.status.

שעונים רבים ברשת משתמשים בתמונות המשתנות דינמית על פי שעון המחשב. קוד כזה מחייב בדיקה חוזרת של מחרוזת השעה והתאמת התמונות במקום הנכון. ניישם שעון ב**שורת המצב** (status bar).

בכל ריצה של הפונקציה יש לייצר את אובייקט התאריך, מכיון שאובייקט התאריך הוא למעשה מחרוזת המייצגת את הזמן המדויק ברגע יצירתו בלבד.

```
function makeClock() {
    thisDate = new Date() ;
```

לאחר מכן, יש ליצור מחרוזת שתכיל את השעות, הדקות והשניות מתוך אובייקט התאריך, האובייקט Date.

```
strDate = thisDate.getHours() + ":"  ;
strDate = strDate + thisDate.getMinutes() + ":" ;
strDate = strDate + thisDate.getSeconds() ;
```

לבסוף, כל מה שנותר הוא להציב את המחרוזת strDate בשורת המצב.

```
 window.status = strDate ;
}
```

הפונקציה ()makeClock מציגה את השעון פעם אחת. כדי להפעיל את הפונקציה ()makeClock, יש להפעיל אותה בעזרת השיטה ()setInterval. בפונקציה ()initClock יש משפט אחד המפעיל את הפונקציה ()makeClock בעזרת השיטה ()setInterval, באופן הבא:

```
function initClock() {
    setInterval('makeClock();',1000) ;
   }
```

המספר 1000 בפונקציה ()setInterval מפעיל את הפונקציה ()makeClock כל 1000 מילישניות. הקריאה לפונקציה המבצעת, הפונקציה ()initClock, נמצאת בתגית <body> במתכונת של אירוע (event-handler) בשם onload.

לפניך הקוד המלא להצגת שעון בשורת המצב.

דוגמה 8-02.html:

```
<html dir="rtl">
 <head>
  <title>8-02</title>
  <script language="javascript">
   function initClock() {
    setInterval('makeClock();',1000) ;
   }

   function makeClock() {
    thisDate = new Date() ;
    strDate = thisDate.getHours() + ":"  ;
    strDate = strDate + thisDate.getMinutes() + ":" ;
    strDate = strDate + thisDate.getSeconds() ;
    window.status = strDate ;
   }
  </script>
 </head>
 <body onload="initClock();">
   <h3>
     השעון מתעדכן בשורת המצב
   </h3>
 </body>
</html>
```

שים לב לשורת המצב בתרשים המסך:

תרשים 8.2: שים לב לתאריך המוצג בשורת המצב.

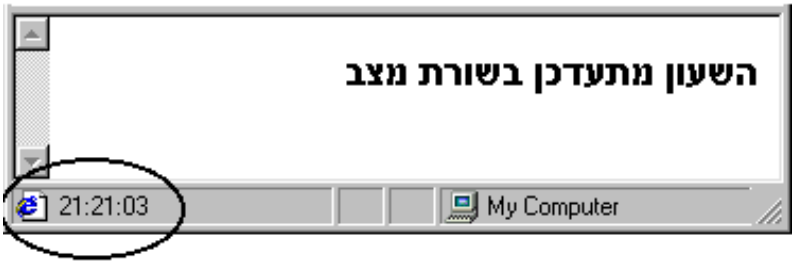
כדאי שוב לשים לב להפעלת השיטה ()setInterval, המופעלת מתוך הפונקציה ()initClock ומפעילה את הפונקציה ()makeClock. לחילופין, ניתן היה לכתוב כך את הפעלת הפונקציה ()makeClock:

```
<body onload="setInterval('makeClock();',1000);">
```

אם הבחנת, השעון שבדוגמה מראה תאריך במבנה hh:mm:ss, אבל מה קורה כאשר הערך להצגה הוא עם ספרה אחת? איך תוצג השעה שמונה בבוקר, 5 דקות ו- 7 שניות? כך : 08:05:07 או 8:5:7?

בדוגמה **8-03.html** תמצא קוד משופר המציג את השעון בשורת המצב (status bar) :

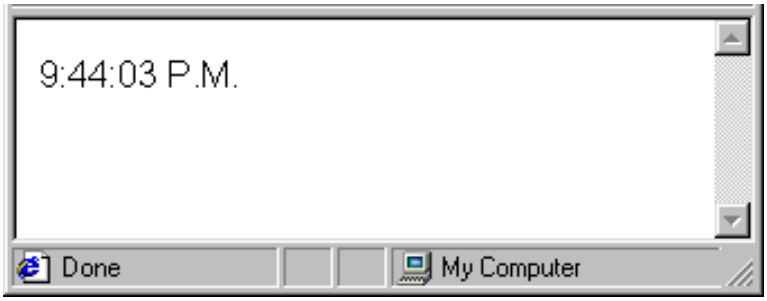
```
<html dir="rtl">
 <head>
  <title>8-03</title>
  <script language="javascript">
   function makeClock() {
    thisDate = new Date() ;
    hh = thisDate.getHours() ;
    mm = thisDate.getMinutes() ;
    ss = thisDate.getSeconds() ;
    if (hh < 10) hh = "0" + hh ;
    if (mm < 10) mm = "0" + mm ;
    if (ss < 10) ss = "0" + ss ;
    strDate = hh + ":" + mm + ":" + ss ;
    window.status = strDate ;
   }
  </script>
 </head>
 <body onload="setInterval('makeClock();',1000);">
  <h3>
   השעון מתעדכן בשורת מצב
  </h3>
 </body>
</html>
```



תרשים 8.4 : שעון משופר המציג אפס מוביל לפני שנייה, דקה או שעה הקטנים מ- 10.

הצגת שעון כחלק מדף HTML

שורת המצב אינה המקום האידיאלי להצגת השעון. סביר יותר להציג את השעון בחלק מדף HTML. קובץ **showClock.html** לקוח מהספר **JavaScript+CSS+DOM למפתחי אתרים באינטרנט** ולכן ייתכן שחלק מהקוד לא מובן לך. אתה בוודאי תזהה את השיטות שהשתמשת בהן קודם לכן. החידוש בקובץ **showClock.html** הוא בשינוי דינמי של תוכן בדף HTML, במקרה זה הצגת השעה.



תרשים 8.5: הצגת השעון כחלק מדף HTML.

תאריך, שעון וטופס

שדה תאריך או שעה מופיע בטופס כשדה מסוג select ולא כשדה מסוג text. בשדה מסוג טקסט יכול הגולש להקליד את אותו תאריך בסגנונות שונים. נניח והגולש יתבקש להקליד בשדה טקסט שבטופס את תאריך יום ההולדת שלו שהוא 10 במאי 1988. להלן האפשרויות השונות: 10-05-1988, 10/5/88, 5/10/1988, 10-May-88 ויש עוד. גם אם תבקש מהמשתמש להקליד את התאריך במבנה מסוים, למשל, dd-mm-yyyy בכלל לא בטוח שהוא ינהג לפי בקשתך. לכן:

- כאשר הטופס מחייב הכנסת תאריך על ידי הגולש, עליך להגדיר למשתמש 3 שדות מסוג select לבחירת היום, החודש והשנה. מכיון שכך, בדיקות התקינות יוודאו שהמשתמש לא בחר את היום ה- 31 בחודש שיש בו רק 30 יום.
- כאשר הטופס מחייב הכנסת תאריך על ידי המערכת, עשה שימוש באובייקט Date.

עוד על אובייקט התאריך

השיטה **getTime()** מחזירה מחרוזת המייצגת את מספר אלפיות השנייה שחלפו מאז 1.1.1970. לדוגמה, 886774180650.

השיטה **setTime()** קובעת את התאריך לפי הנוסחה: 1.1.1970 ועוד מספר אלפיות השנייה המוזנות בין הסוגריים של השיטה, אשר מתורגמות לימים.

כדי לקבל את התאריך הנמוך ביותר:

```
todaysDate = new Date() ;
todaysDate.setTime(0) ;
```

והערך שיתקבל במשתנה todaysDate יהיה:

```
Thu Jan 01 02:00:00 UTC+0200 1970
```

הדוגמה הבאה מחשבת תאריך המייצג חודש אחד לאחר התאריך הנוכחי. עקוב אחר הפעולות:

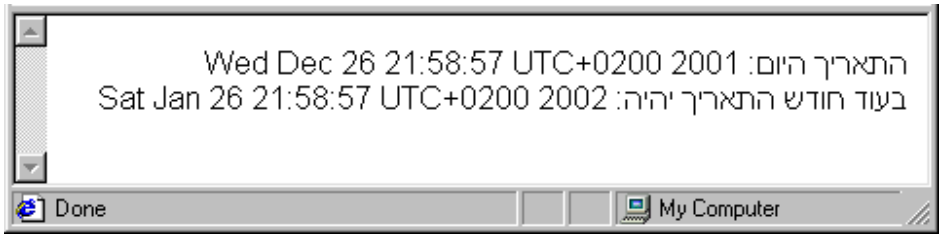
1. הגדר אובייקט תאריך המכיל את התאריך הנוכחי.
2. שמור את התאריך כמחרוזת של אלפיות שנייה על ידי שימוש בשיטה **getTime()**.
3. צור מחרוזת המייצגת אלפיות שנייה בחודש (לפי 31 יום בחודש):
   א. שנייה אחת שווה 1.
   ב. כפול ב- 1000 אלפיות שנייה ב-1, כדי להציג שנייה אחת.
   ג. כפול ב- 60 כדי להציג דקה אחת.
   ד. כפול ב- 60 כדי להציג שעה אחת.
   ה. כפול ב- 24 כדי להציג יום אחד.
   ו. כפול ב- 31 כדי להציג חודש אחד (בערך).
4. חבר את התוצאה של סעיף 2 וסעיף 3 (מה שהתקבל מייצג את מספר אלפיות השנייה שחלפו מ- 1.1.1970 ועד חודש קדימה מהתאריך הנוכחי).
5. קבע את ערך אובייקט התאריך למחרוזת החדשה שיצרת בסעיף 4 בעזרת setTime().
6. הצג את התאריך.

דוגמה 8-04.html :

```
<html dir="rtl">
 <head>
  <title>8-04</title>
  <script language="javascript">
   todayDate = new Date() ;
   document.write("התאריך היום: " + todayDate) ;
   newDate = todayDate.getTime() + (1*1000*60*60*24*31) ;
   todayDate.setTime(newDate) ;
   document.write("<br />")
   document.write("בעוד חודש התאריך יהיה: ") ;
   document.write(todayDate) ;
  </script>
 </head>
</html>
```

השימוש בשתי שיטות אלו : setTime() ו- getTime() נעשה לחישוב תאריכים מרוחקים לעתיד או בעבר.

התוצאה בדפדפן תוצג כך :



תרשים 8.3 : ביצוע חישובים על התאריך.

שאלה 8.1

הצג שעון של 12 שעות בשורת המצב (status bar). את השעה 08:00 הצג 08:00AM ואת השעה 20:30 הצג 08:30PM.

פרויקט סיינפלד

אובייקט התאריך, Date, הוא אחד מהאובייקטים של JavaScript שאין בו הרבה שימוש (זו דעתי). יש אתרים שמשתמשים באובייקט זה כדי להציג את השעה לגולש (כאילו שהוא לא יודע). מה עוד שבמערכת ההפעלה Windows השעה היא חלק (כמעט בלתי נפרד) משורת המשימות. מי שהשעה לא מופיעה לו בשורת המשימות - שיקום! אף אחד לא קם (מתוך לקט מערכונים ושירים של הגשש החיוור).

נוסיף לטופס ההרשמה למועדון את היום והשעה בעת מילוי הטופס (למעשה ברגע שהמשתמש לחץ על לחצן **שלח טופס**). לשם כך, נציג את התאריך והשעה בשדה שלא ניתן לשנותו (readonly).

כך נגדיר את השדות החדשים בטופס:

```
<input name="inputDate" type="text" size="10" readonly="readonly" />
<input name="inputTime" type="text" size="8" readonly="readonly" />
```

וכך נראית הפונקציה:

```
function putDateTime()  {
  today = new Date() ;
  dd = today.getDate() ;
  dd = addZero(dd) ;
  mm = today.getMonth() + 1 ;
  mm = addZero(mm) ;
  yyyy = today.getFullYear() ;
  ddmmyyyy = dd + "/" + mm + "/" + yyyy ;
  window.document.seinfeldForm.inputDate.value = ddmmyyyy ;

  hh = today.getHours() ;
  hh = addZero(hh) ;
  ss = today.getSeconds() ;
  ss = addZero(ss) ;
  hhss = hh + ":" + ss ;
  window.document.seinfeldForm.inputTime.value = hhss ;
}

function addZero(unit)  {
 if (unit < 10) unit = '0' + unit ;
 return unit;
}
```

פונקציית העזר, addZero(), נועדה להוסיף 0 מוביל. היא בנויה כך שתוכל לקבל כל מספר שהוא: יום, חודש, שעה או דקה.

הפעלת הפונקציה:

```
<form name="seinfeldForm"
   onsubmit="putDateTime(); return checkForm();">
```

פרק 9

מסגרות

טכנולוגיית **המסגרות** (frames) מאפשרת הצגת מספר דפי HTML על מסך אחד. למרות שנושא זה נסקר בחלק הראשון של הספר **מבוא לתכנות בסביבת אינטרנט, מבוא ו-HTML**, אחזור בקצרה על הנושא. אם אתה זקוק לתזכורת בנושא מסגרות, רצוי שתחזור לעיין בחלק הראשון של הספר **מבוא לתכנות בסביבת אינטרנט, מבוא ו-HTML**.

<frameset> - הפקודה פותחת קבוצת מסגרות. לפקודה זו מספר מאפיינים, כגון rows ו-cols הקובעים אם החלוקה היא לשורות או לטורים, או המאפיין frameborder המאפשר ביטול הגבולות בין המסגרות.

<frame /> - הפקודה מגדירה את תכונות המסגרת עצמה וכוללת מאפיינים, כגון src, הקובעים איזה מסמך יופיע במסגרת. המאפיין name, למשל, מאפשר לקבוע שם למסגרת, כדי שאפשר יהיה להתייחס אליה בזמן ריצה.

דוגמה:

```
<frameset rows="20%,*">
    <frame name="menuBar" src="menuTOP.html" />
    <frame name="content" src="productList.html" />
</frameset>
```

הפעלת קישור בין מסגרות מתבצעת על ידי שימוש במאפיין **target** של התגית **<a>**. דוגמה לקישור למסגרת בשם content ניתן לראות בקטע הקוד הבא:

```
<a href="http://www.hod-ami.co.il" target="content">
הוצאת הוד-עמי
</a>
```

טיפ

רצוי לתת למסגרת שם משמעותי. אם המסגרת תכיל תוכן קרא לה content, main או conFrame. אם המסגרת מכילה תפריט קרא לה menuBar, topFrame, upperMenu או bigMenu. אם המסגרת מכילה פרסומת קרא לה בשם שירמוז לך על מה שמוצג בתוכה.

חומר עזר

חומר לימוד בנושא מסגרות (frames) תוכל למצוא בספר **HTML 4 למפתחי אתרים באינטרנט** בהוצאת **הוד-עמי**.

שמירת נתונים במסגרות סטטיות

אחד השימושים העיקריים בטכנולוגיית המסגרות הוא חלוקת החלון למספר מסגרות. קיימת מסגרת סטטית המכילה את התפריט ומסגרות דינמיות המכילות תוכן, בהתאם לבחירה שבתפריט. ניתן להשתמש במסגרות הסטטיות, כדי לשמור נתונים לאורך כל זמן השימוש באתר. התגית **<frameset>** מגדירה אובייקט בשם parent. כל התייחסות למסגרת שהוגדרה בעזרת התגית **<frame />** תחת התגית **<frameset>** תהיה דרך אובייקט **parent**. ההתייחסות לתוכן המסגרת תהיה:

```
parent.frameName.document
```

בדוגמה הבאה מודגמת מערכת מסגרות המחלקת את המסך לשתי עמודות. שמה של המסגרת הימנית הוא menuBar והיא תכיל את המסמך **menu.html**. שמה של המסגרת השמאלית הוא mainScreen והיא תכיל את המסמך **first.html**.

mainScreen	menuBar

דוגמה 9-01.html: תבנית המסגרות

```
<html>
  <head>
    <title>9-01</title>
  </head>
  <frameset cols="50%, *">
    <frame name="mainScreen" src="first.html" />
    <frame name="menuBar" src="menu.html" />
  </frameset>
</html>
```

דף **menu.html** יכיל קישורים לשני מסמכים: **first.html** ו-**second.html**, כך שייטענו במסגרת השמאלית "target="mainScreen.

במסמך **menu.html** ישנו גם **טופס** (form) בשם myForm, אשר יאפשר הצגת שני מונים: firstCounter ו-secondCounter. מונים אלה יכילו את מספר הפעמים שהופעלו הקישורים בדף **menu.html**, כדי לראות את כל אחד משני הדפים שבאתר. בשדה

נוסף בשם currentPage, יוצג שם הדף הנוכחי המוצג באותה עת במסגרת השמאלית הדינמית.

בתסריט JavaScript שבקובץ **menu.html** הוגדרו שני משתנים (firstCounterVar, secondCounterVar), אשר יכילו את מספר הפעמים שהמבקר נמצא בדפים המשניים. הפונקציה counterDisplay() (גם היא בדף **menu.html**) מציגה את ערכי המשתנים בשדות הטקסט אשר בדף, אולם אין במסמך זה (menu.html) אף קריאה לפונקציה. הקריאה לפונקציה תתבצע מהדפים האחרים: first.html ו-second.html, בזמן שהם ייטענו. קצת מורכב? מייד הדברים יתבהרו.

קובץ **menu.html**: מסמך התפריט ימוקם במסגרת הסטטית menuBar.

```
<html dir="rtl">
  <head>
   <title>menu</title>
   <script language="javascript">
    var firstCounterVar = 0 ;
    var secondCounterVar = 0 ;
    function counterDisplay() {
     document.myForm.firstCounter.value = firstCounterVar ;
     document.myForm.secondCounter.value = secondCounterVar ;
    }
  </script>
 </head>
 <body bgcolor="black" text="white" link="white" vlink="white">
  <font size="5">
   תפריט
  </font>
  <hr />
  <a href="first.html" target="mainScreen">עמוד ראשון</a>
  <br />
  <a href="second.html" target="mainScreen">עמוד שני</a>
  <hr />
  <form name="myForm">
   מספר הפעמים שנראה העמוד הראשון:
   <input type="text" name="firstCounter" size="3" value="0" />
   <br />
   מספר הפעמים שנראה העמוד השני:
   <input type="text" name="secondCounter" size="3" value="0" />
   <br />
   העמוד הנוכחי:
   <input type="text" name="currentPage" />
  </form>
 </body>
</html>
```

במסגרת השמאלית הדינמית יופיע דף first.html או דף second.html. שני הדפים מעדכנים את משתני המונים (המשתנים firstCounterVar, secondCounterVar), הנמצאים במסמך menu.html שבמסגרת הסטטית menuBar. הפנייה למשתנה שנמצא במסגרת אחרת מתבצעת בעזרת המילה parent, המציינת כי יש לפנות להורה המסגרת שנמצא ברמה אחת גבוהה יותר. לאחר שעלינו רמה אחת, ניתן לפנות למסגרת הרלוונטית menuBar ולמשתנה שבה, firstCounterVar.

התחביר המלא הוא:

```
parent.menuBar.firstCounterVar += 1 ;
```

כעת, נקבע את הערך המתאים לשדה currentPage במסגרת menuBar על ידי המשפט:

```
parent.menuBar.document.myForm.currentPage.value = "עמוד ראשון" ;
```

נקרא לפונקציה menuBar.counterDisplay() על ידי שימוש באותו תחביר:

```
parent.menuBar.counterDisplay() ;
```

קובץ **first.html**. מסמך דף הבית ימוקם במסגרת הדינמית mainScreen:

```
<html dir="rtl">
 <head>
  <script language="javascript">
    parent.menuBar.firstCounterVar += 1 ;
    parent.menuBar.document.myForm.currentPage.value = "עמוד ראשון" ;
    parent.menuBar.counterDisplay() ;
  </script>
  <title>first</title>
 </head>
<body bgcolor="white">
  <font size="5">
  עמוד ראשון
  </font>
</body>
</html>
```

קובץ **second.html** שיופיע באתר במסגרת הדינמית mainScreen:

```
<html dir="rtl">
  <head>
    <script language="javascript">
      parent.menuBar.secondCounterVar += 1 ;
      parent.menuBar.document.myForm.currentPage.value = "עמוד שני" ;
      parent.menuBar.counterDisplay() ;
    </script>
    <title>second</title>
```

```
  </head>
  <body bgcolor="yellow">
  <font size="5">
   עמוד שני
  </font>
  </body>
</html>
```

לפניך התוצאה בדפדפן:

תרשים 9.1: מראה המסגרות בעת טעינה.

וזו התוצאה לאחר לחיצה על הקישור **עמוד שני**.

תרשים 9.2: הפעלת קישור במסגרת הימנית מציגה דף מתאים במסגרת השמאלית ומוסיפה 1 למונה המתאים שבמסגרת הימנית.

חומר עזר

עוד על עבודה עם מסגרות (frames) תוכל למצוא בספר **JavaScript סדנת לימוד** בהוצאת **הוד-עמי**, פרק 28.

במסגרות שבנית לעיל יש בעיות. דף first.html נטען למסגרת השמאלית mainScreen. להזכירך, בדף first.html יש תסריט JavaScript שמעדכן את שדות הטופס במסגרת הימנית וזה לא קורה - עובדה! המונה של העמוד הראשון נשאר 0. במקום לקבל במונה העמוד הראשון את המספר 1, התקבלה הודעת שגיאה. אתה יכול להבחין בהודעת שגיאה בשורת המצב שבתחתית החלון (נכון שהודעת השגיאה אינה קריטית במשמעות שניתן להמשיך ולעבוד, אבל זה לא יפה). לחיצה כפולה עליה תגלה לך את המשפט הבא:

```
parent.menuBar.document.myForm.currentPage is not an object
```

באותו זמן נטען דף menu.html למסגרת הימנית menuBar, וערך שני המונים נקבע ל-0. שאל את עצמך את השאלות הבאות ונסה לאמת את תשובותיי:

- האם המסגרת menuBar קיימת? כן
- האם מוצג בתוכה דף בשם menu.html? כן
- האם בדף menu.html מוגדר טופס בשם myForm? כן
- האם בדף menu.html מוגדר טופס בשם myForm ובתוכו מוגדר שדה טקסט בשם currentPage? כן
- אז מה הבעיה?!?

נתקלת עכשיו באחת הבעיות בעבודה עם מסגרות. כל עוד עבדת עם מסגרות שהקשר היחיד ביניהן היה באמצעות קישורים בלבד - לא היתה בעיה והמסגרות "סיפקו את הסחורה". כעת, משנוסף תסריט JavaScript הפונה באמצעות ה-DOM ממסגרת אחת למסגרת אחרת נוצרה בעיה. הבעיה נובעת מתהליך טעינת הדפים לתוך המסגרת. במקרה שלפנינו, קובץ **9-01.html** מגדיר את המסגרות וטוען את הדפים לתוכן. דף first.html נטען לפני דף menu.html וגם מתחיל להתבצע לפניו. כך, שכאשר מופעל התסריט בדף first.html המתייחס לשדות טופס המוגדרים בדף menu.html (שעדיין לא נטען או עדיין לא הסתיימה טעינתו), מתקבלת הודעת שגיאה, מכיון שהאובייקט אליו פנה התסריט טרם הוגדר.

בעיית התזמון היא בעיה קריטית בעת עבודה עם מסגרות. צריך לוודא שאכן המסגרת קיימת ואכן הדף המתאים נמצא בתוכה.

אחד הפתרונות לבעיית התזמון הוא ליצור דף נוסף שייטען ראשון למסגרת mainScreen ושאין בו תסריט המתייחס למסגרות אחרות. באופן זה, דף menu.html יספיק להיטען במסגרת menuBar. כעת, הפעלת קישור במסגרת הימנית תציג את הדף המתאים במסגרת השמאלית. אותו דף במסגרת השמאלית יפעל על שדות הטופס הנמצאים במסגרת הימנית ללא הודעת שגיאה.

ראה **דוגמה 9-02.html**:

```
<html>
  <head>
    <title>9-02</title>
  </head>
  <frameset cols="50%, *">
    <frame name="mainScreen" src="homePage.html" />
    <frame name="menuBar" src="menu.html" />
  </frameset>
</html>
```

שאלה 9.1

בנה דף עם שתי מסגרות: עליונה ותחתונה. במסגרת העליונה הצג שדה טופס מסוג select בו ניתן לבחור את דגם מטוס אל-על. לפי הבחירה תוצג התמונה במסגרת התחתונה. את תמונות מטוסי אל-על תמצא בתיקיה images של הקבצים לפרק זה, הניתנים להורדה מאתר האינטרנט של הספר בכתובת www.hod-ami.co.il/59355.html.

שאלה 9.2

סדרת התמונות הבאה משמשת להצגת 5 לחצנים (כל הקבצים הנדרשים לתרגיל זה ניתנים להורדה מאתר האינטרנט של הספר בכתובת www.hod-ami.co.il/59355.html). בהתחלה מוצגים כל הלחצנים (תמונות) שהאות האחרונה היא L. כאשר העכבר נמצא על הלחצן, מתחלפת התמונה של הלחצן לאותו שם תמונה עם סיומת R. לדוגמה, כאשר הסמן יהיה על התמונה internetL.gif היא תתחלף לתמונה internetR.gif.

דף מקושר	<picture>L.gif	<picture>R.gif
Internet.htm	internetL.gif	internetR.gif
Lang.htm	progL.gif	progR.gif
MCSE2000.htm	mcseL.gif	mcseR.gif
PCtech.htm	pctechL.gif	pctechR.gif
Office2k.htm	office2kL.gif	office2kR.gif

א. בנה דף ובו שתי מסגרות אנכיות.

ב. מקם את חמשת הלחצנים במסגרת הימנית.

ג. במסגרת השמאלית הצג את הדף Hod-Ami.html.

ד. לחיצה על לחצן תציג את הדף המתאים במסגרת השמאלית.

ה. הצב לחצן נוסף במסגרת הימנית ועליו הכותרת "דף הבית". לחיצה על לחצן זה תציג את דף Hod-Ami.html במסגרת הימנית.

ו. הצב לחצן נוסף במסגרת הימנית ועליו הכותרת "סיום". לחיצה על לחצן זה תציג בעזרת השיטה ()alert את מספר הפעמים שהמשתמש לחץ על כל לחצן במסגרת הימנית.

כמה טיפים על מסגרות

לצד היתרונות של עבודה במסגרות, יש לציין גם את החסרונות. הסיבה הבולטת היא בעיות בשמירת רשימות **מועדפים** (Favorites), בעיות בהדפסה ואיטיות מסוימת בטעינת הדפים הראשונים. למרות זאת, טכנולוגיה זו נוחה מאוד.

בעיה נוספת עלולה להתעורר עקב הודעות שגיאה הנובעות מבדיקת משתנים או ערכים, הנמצאים במסגרות שונות שטרם נטענו או הושלמה טעינתן אל הדפדפן. הודעות שגיאה אלו לרוב אינן מפריעות לריצת הקוד, אך הימצאותן אינה מעידה על מקצוענות.

חומר עזר

עוד על עבודה במסגרות תוכל לקרוא בספר **עיצוב ממשק באינטרנט** בהוצאת **הוד-עמי**.

פרויקט סיינפלד

אחד מכלי הניווט בעבודה עם מסגרות היה ההפרדה בין מסגרת הניווט בה נמצאים הקישורים לבין מסגרת התוכן. בהתחלה הקישורים היו טקסט ואחר כך תמונה, אבל קישור זה קישור. כלי נוסף לניווט הוא הרשימה הנגללת. כן, זהו שדה טופס ובו רשימה, כאשר בחירה מהרשימה מציגה במסגרת התוכן את הדף המתאים.

בדף menu.html (של פרויקט סיינפלד) נבנה שדה טופס שכזה:

```
<form name="navBar">
  <select name="selLink">
   <option value="0">&lt;&lt; בחר נושא&gt;&gt;</option>
   <option value="1">דף הבית</option>
   <option value="2">אודות</option>
   <option value="3">החברות של ג'ריי</option>
   <option value="4">מועדון</option>
   <option value="5">חידון</option>
   <option value="6">מילון</option>
  </select>
</form>
```

כאשר ייבחר ערך מרשימה זו, אז ערכו יישמר בתכונה:

```
document.navBar.selLink.selectedItem
```

הערך, כפי שהוגדר במאפיין value של אפשרות כלשהי יהיה:

```
document.navBar.selLink.options[n].value
```

הערך של האפשרות שנבחרה, כפי שהוגדרה במאפיין value יהיה:

```
document.navBar.selLink.options[document.navBar.selLink.selectedItem].value
```

נממש זאת בפונקציה link2go(), באופן הבא:

```
function link2go()  {
 iChoose = document.navBar.selLink.selectedIndex ;
 iValue = document.navBar.selLink.options[iChoose].value ;
 iValue = parseInt(iValue) ;
 switch (iValue) {
   case 1:
     ilink = "homePage.html" ;
     break ;
   case 2:
     ilink = "About.html" ;
     break ;
   case 3:
     ilink = "Girls.html" ;
```

```
      break ;
    case 4:
      ilink = "Club.html" ;
      break ;
    case 5:
      ilink = "Trivia.html" ;
      break ;
    case 6:
      ilink = "Yada.html" ;
      break ;
  }
  parent.content.location = ilink ;
}
```

המשפט המרכזי בפונקציה link2go() הוא הצבת הקישור המתאים בתכונה location של מסגרת התוכן. להזכירך, למסגרת התוכן קוראים... content וזה הוגדר בקובץ **seinfeldFrame.html**.

שאלה: למי שייכת התכונה location? לאובייקט window או לאובייקט document?

תשובה: לאובייקט window, לכן היית יכול גם לכתוב:

```
window.parent.content.location = ilink ;
```

ועכשיו, איך מקשרים בין שדה select לפונקציה link2go(). מכיון שאנו רוצים שהדף יוצג בעת בחירת האפשרות בשדה, יהיה עלינו לתפוס את האירוע onchange באופן הבא:

```
<select name="selLink" onchange="link2go();">
```

פרק 10

חלונות

בעזרת JavaScript אפשר לפתוח חלונות נוספים של הדפדפן ובכך להעשיר את ממשק המשתמש. חשוב לזכור כי חלונות דפדפן נוספים אינם תמיד חביבים על המשתמש, וניהול לא נכון שלהם עלול לגרום לבלבול. קרא על כך בספר **עיצוב ממשק באינטרנט** בהוצאת **הוד-עמי**.

למשל, הדף butter.html ייפתח בחלון דפדפן חדש:

```
<a href="butter.html" target="_blank">ללא כולסטרול</a>
```

אפשר גם לפתוח חלון ריק, כך:

```
window.open() ;
```

אבל, מה הטעם לפתוח חלון ריק?

יש טעם! אבל, כדי שתוכל לעבוד עם החלון הריק צריך שיהיה לו שם (name), אחרת לא תוכל להתייחס אליו ואל האובייקטים שבו. Javascript מציעה, מעבר לפתיחת חלון, גם יכולת להתייחס אליו ואל האובייקטים שבו (בין אם הוא חלון ריק או חלון שיש בו תוכן של דף HTML).

בדוגמה הבאה מוגדר המשתנה newWin כאובייקט חלון חדש:

```
newWin = window.open("","testWindow","width=200, height=200") ;
```

אם תרשום שורת קוד זו במסמך HTML, ייפתח חלון חדש וריק. שם החלון הוא testWindow והוא נפתח ברוחב 200 נקודות ובגובה 200 נקודות. הוא ימוקם על המסך היכן שהדפדפן יחליט.

המשתנה newWin הוא משתנה מסוג אובייקט ומייצג חלון בשם testWindow. למעשה, יש לחלון החדש שנפתח שני שמות:

newWin ישמש תסריטי JavaScript כדי לגשת לאובייקטים שבחלון. למשל, כדי לשנות את צבע הרקע של המסמך שבחלון החדש שנפתח, יהיה עליך לכתוב (במסמך אחר):

```
newWin.document.bgColor = "red" ;
```

השם **testWindow** ישמש לגישה דרך המאפיין action בתגית <form>, או המאפיין target בתגית <a>. לאחר שנפתח את החלון החדש בשם testWindow, תוכל לרשום (במסמך אחר):

```
<a href="myClub.html" target="testWindow">מועדון לקוחות</a>
```

להלן תחביר השיטה window.open():

```
object = window.open("content","window name","window attributes") ;
```

כפי שבוודאי שמת לב, החלון שנוצר בעקבות הפעלת התסריט:

```
newWin = window.open("","testWindow","width=200, height=200") ;
```

הוא חלון בסיסי שאינו מכיל לחצנים או פסי גלילה. ניתן להעניק לחלון חדש מאפיינים רבים באמצעות פרמטרים הנכתבים בסוגריים ובין גרשיים, בהפעלת השיטה window.open().

להלן רשימת המאפיינים והפרמטרים האפשריים. שים לב שבמקום yes או no, אפשר לכתוב 1 או 0 בהתאמה (ברירת המחדל מסומנת בהדגשה):

מאפיין	משמעות	ערכים
width	רוחב החלון	n pixels
height	גובה החלון	n pixels
top	מרחק החלון משמאל המסך	n pixels
left	מרחק השדה מראש המסך	n pixels
resizable	אפשרות שינוי גודל	no, **yes**
scrollbars	פסי גלילה	no, **yes**
menubar	שורת תפריטים	no, **yes**
toolbar	שורת לחצנים	no, **yes**
status	שורת מצב	no, **yes**
location	שורת מיקום	no, **yes**
fullscreen	מסך מלא	**no**, yes
directories	סרגל Links	no, **yes**
titlebar	שורת כותרת	no, **yes**

דוגמה לפתיחת חלון באמצעות תסריט:

```
myNewWindow = window.open("about.html","allAbout","width=100,
↳ height=100, resizable=yes, toolbar=yes") ;
```

אם לא יצוין ערך לפרמטר (yes או no ובהתאמה: 1 או 0), ייקבע ערך ברירת המחדל no או yes.

דרך אחרת לפתיחת חלון בהפעלת קישור:

```
<a href="javascript:void(0);"
onclick="myNewWindow = window.open('about.html', 'allAbout',
↳'width=100, height=100, resizable=yes, toolbar=no') ;">
bla bla bla</a>
```

קובץ **10-01.html** מרכז מספר אפשרויות לפתיחת קובץ.

דוגמה 10-01.html:

```
<html dir="rtl">
 <head>
  <title>10-01</title>
  <script language="javascript">
   function NewAndEmptyNoName() {
    window.open() ;
   }

   function NewWithName(what2open) {
    mySon = window.open(what2open) ;
   }
  </script>
 </head>
<body>
 <h3>
  פתיחת חלון חדש
 </h3>

 <a href="javascript:void(0);"
   onclick="window.open('amit.html');">
  פתיחת חלון חדש ללא שם ובתוכו דף קיים
 </a>
 <br />

 <a href="javascript:void(0);"
   onclick="newWin=window.open('amit.html');">
  פתיחת חלון חדש עם שם ובתוכו דף קיים
 </a>
 <br />

 <a href="javascript:void(0);"
   onclick="NewAndEmptyNoName();">
  פתיחת חלון חדש ללא שם וריק באמצעות פונקציה
 </a>
 <br />
```

```
 <a href="javascript:void(0);"
   onclick="NewWithName('amit.html');">
  פתיחת חלון חדש עם שם ובתוכו דף קיים באמצעות פונקציה
 </a>
 <br />
</body>
</html>
```

הפעל את הדף ואת הקישורים שבו וראה מה שמתרחש. הפונקציה **NewWithName()** פותחת חלון חדש עם תוכן הקובץ ששמו הועבר אליה כפרמטר. מבחינה ויזואלית, הקישור הראשון, השני והרביעי מציגים את אותו דבר. אבל, הקישור השני והרביעי נותנים שם לחלון החדש, דבר שתוכל להתייחס אליו בהמשך.

◆ שאלה 10.1

בנה דף עם 5 לחצנים/קישורים (כל החומרים הנדרשים ניתנים להורדה מאתר האינטרנט שלנו בכתובת www.hod-ami.co.il/59355.html):

א. הלחצן הראשון יקשר לקובץ episode01.html.

ב. הלחצן השני יקשר לקובץ episode02.html וכך הלאה.

ג. קבצי episode*.html ייפתחו בחלון נפרד בעזרת פונקציה שתבנה במיוחד לצורך כך (כל הפעלת קישור תציג את הדף בחלון חדש ונפרד).

ד. תן ללחצנים/קישורים שמות מתאימים בהתאם לתוכן הקבצים שהם פותחים.

בתיקיה images תמצא את תמונת צוות השחקנים בשם cast.jpg ותוכל להיעזר בה.

◆ שאלה 10.2

בהמשך לשאלה הקודמת, הוסף לחצן שיסגור את כל החלונות שנפתחו.

חומר עזר

עוד על חלונות קרא בספר **JavaScript סדנת לימוד**, פרק 10, בהוצאת **הוד-עמי**, ובספר **JavaScript+CSS+DOM למפתחי אתרים באינטרנט**, פרק 5, בהוצאת **הוד-עמי**.

העברת מידע בין חלונות

בדומה לעבודה עם **מסגרות** (frames), ניתן לפנות לפונקציות ולערכי משתנים בחלונות שונים תוך שימוש בשם משתנה החלון. כאשר נפתח חלון: newWin=window.open(...), נוכל לפנות אל פונקציות שנמצאות בו על ידי פקודה במבנה הבא:

```
newWin.functionName
```

נוכל גם לשנות מאפיינים, כגון צבע רקע, על ידי פקודה בתחביר הזה:

```
newWin.document.bgColor = 'red' ;
```

קובץ **10-02.html** מדגים פתיחת חלון, שינוי מאפיינים והפעלת שיטות מהחלון שפתח אותו.

דוגמה 10-02.html:

```
<html dir="rtl">
 <head>
  <title>10-02</title>
  <script language="javascript">
   function NewWithName(what2open) {
    myFeatures = 'directories=no,fullscreen=no,location=no,menubar=yes,' ;
    myFeatures += 'status=yes,titlebar=no,toolbar=no,' ;
    myFeatures += 'scrollbars=yes,width=280,height=302,resizeable=yes'
    mySon = window.open(what2open, 'bigOne', myFeatures) ;
    mySon.focus() ;
   }
   function paintRed() {
    mySon.document.bgColor = "red" ;
    mySon.focus() ;
   }

   function changeContent() {
    mySon.document.location = "eyal.html" ;
    mySon.focus() ;
   }
  </script>
 </head>
```

```
<body>
 <a href="javascript:void(0);"
   onclick="NewWithName('amit.html');">
  פתיחת חלון חדש באמצעות פונקציה:
 </a>
 <br />

<ul>
 <li>
  <a href="javascript:void(0);" onclick="paintRed();">
   שנה צבע רקע לאדום
  </a>
  <br />
 </li>
 <li>
  <a href="javascript:void(0);" onclick="changeContent();">
   שנה תוכן בחלון
  </a>
  <br />
 </li>
 <li>
  <a href="javascript:void(0);" onclick="mySon.close();">
   סגור חלון
  </a>
  <br />
 </li>
</ul>
</body>
</html>
```

כדי לראות איך באמת זה עובד, יהיה עליך להפעיל את הקובץ **10-02.html** ולהפעיל את הקישור **פתיחת חלון חדש באמצעות פונקציה** (חובה להפעיל קישור זה, אחרת כל שאר הקישורים לא יעבדו, מכיון שלא קיים החלון אליו הם מתייחסים). לחיצה על אחד הקישורים ברשימת התבליטים תשפיע על מראה החלון שנפתח.

הערה

הסיבה שניתן לגשת ל-mySon שהוגדר בפונקציה NewWithName() גם מפונקציות אחרות, היא ש-mySon הוא אובייקט מסוג חלון ובתור שכזה הוא מוגדר כמשתנה אובייקט גלובלי.

תרשים 10.1 : הפעלת הקישור **פתיחת חלון חדש באמצעות פונקציה**.

כעת הפעל את הקישור **שנה רקע לצבע אדום**.

תרשים 10.2 : הפעל את הקישור **שנה רקע לצבע אדום**.

לחץ על הקישור **שנה תוכן בחלון**:

תרשים 10.3: הפעלת הקישור **שנה תוכן בחלון**.

בעת הפעלת הפונקציה **NewWithName()** בדוגמה **10-02.html**, נבנה משתנה בשם **myFeatures**, בצורה הבאה:

```
myFeatures = 'directories=no,location=no,menubar=yes,' ;
myFeatures += 'status=yes, titlebar=no, toolbar=no,' ;
myFeatures +='scrollbars=yes,width=280,height=302,resizeable=yes' ;
```

משתנה זה מכיל את כל הפרמטרים לפתיחת החלון בעזרת השיטה window.open(). לא חובה לציין את כל הפרמטרים, אך הימצאותם בקוד תקל בעתיד על שינויים שתרצה להפעיל. בשיטת כתיבה "בזבזנית" שכזו כל הפרמטרים נמצאים לנגד עיניך ולא תצטרך להיעזר בספרים כדי למצוא איך קוראים לפרמטר זה או אחר - פתח את הקוד וכל הפרמטרים לפניך.

תוכנית **10-03.html** מדגימה עבודה עם שני חלונות תוך מתן דגש על הנקודות הבאות:

1. מתן גישה לחלון רק אם הוא קיים. בעבודה עם חלונות, יכול להיות מצב בו נרצה לגשת לחלון שנפתח ונסגר. כמובן שניסיון שכזה יציג שגיאה. בתוכנית **10-03.html** נפתח חלון חדש בשם newWin. הפתרון: שימוש במאפיין closed של האובייקט window כדי לדעת אם החלון נסגר.

```
if (newWin.closed) {
 alert("Sorry")
}else{
 newWin.focus() ;
}
```

2. חסימת האפשרות לפתוח חלון נוסף יותר מפעם אחת. לצורך זה נבנה מנגנון המבוסס על גילוי והסתרת שכבה. את הלחצן **פתח חלון חדש** נגדיר כחלק מתגית , באופן הבא:

```
<div id="button2open" style="visibility:visible;">
 <input type="button" value="פתח חלון חדש"
onclick="makeNewWindow();" />
</div>
```

שים לב שבמסגרת המאפיין style של התגית ניתן הערך visible לתכונה visibility. פעולה זו גרמה לכך שהשכבה תיראה וכמובן שתוכל לראות את הלחצן **פתח חלון חדש**.

חומר עזר

נושא שכבות אינו נכלל בתוכנית הלימודים. המעוניינים להרחיב את ידיעותיהם ימצאו חומר מעמיק בספר **JavaScript+CSS+DOM למפתחי אתרים באינטרנט** בהוצאת **הוד-עמי**.

הלחצן **סגור את החלון הנוכחי** תמיד יוצג על גבי המסך.

הלחצנים **עבור לחלון החדש** שנפתח ו- **סגור חלון חדש שנפתח** יוגדרו תחת התגית , באופן הבא:

```
<div id="buttons" style="visibility:hidden;">
 <input type="button" value="עבור לחלון החדש שנפתח"
   onclick="newWinInfocus();" />
 <br /><br />
 <input type="button" value="סגור חלון חדש שנפתח"
   onclick="closeNewWin();" />
 <br /><br />
</div>
```

שים לב שבמסגרת המאפיין style של התגית ניתן הערך hidden לתכונה visibility. פעולה זו גרמה לכך שהשכבה לא תיראה. השכבה תוצג רק כאשר ייפתח חלון חדש, כאשר יופעל הלחצן **פתח חלון חדש**.

כעת יש שתי שכבות שניתן להציגן או להחביאן:

- כאשר מוצגת השכבה button2open המציגה את הלחצן **פתח חלון חדש**, השכבה buttons, המכילה את הלחצנים **עבור לחלון החדש** ו- **סגור חלון חדש שנפתח**, מוחבאת.
- כאשר יופעל הלחצן **פתח חלון חדש**, ניתן יהיה להסתיר את השכבה button2open המכילה את אותו לחצן ולגלות את השכבה buttons המכילה את הלחצנים **עבור לחלון החדש** ו- **סגור חלון חדש שנפתח**.

גילוי והחבאת שכבה ייעשו בעזרת התכונה visibility בפונקציות makeNewWindow() ו- closeNewWin().

חומר עזר

על כל השיטות והתכונות של אובייקט window קרא בספרים הבאים בהוצאת **הוד-עמי**:

JavaScript סדנת לימוד, פרק 10 ובנספח ב'.

JavaScript המדריך השלם, עמודים 965-1043.

JavaScript+CSS+DOM למפתחי אתרים באינטרנט, פרק 5.

ראה כמה "לוגיסטיקה" יש בנושא פתיחת חלונות וכל זאת כדי למנוע שגיאות בעת הרצה. להלן התוכנית המלאה הנמצאת בקובץ **10-03.html**.

דוגמה 10-03.html:

```
<html dir="rtl">
 <head>
  <title>10-03</title>
  <script language="javascript">
   function makeNewWindow()
   {
    myFeatures = 'directories=no,fullscreen=no,location=no,menubar=no,' ;
    myFeatures += 'status=yes,titlebar=no,toolbar=no,' ;
    myFeatures += 'scrollbars=no,width=280,height=302,resizeable=yes' ;
    newWin = window.open('','',myFeatures) ;
    newWin.document.writeln("<html dir='rtl'>") ;
    newWin.document.writeln("<head>") ;
    newWin.document.writeln("<title>The Second HTML page</title>") ;
    newWin.document.writeln("</head>") ;
```

```
    newWin.document.writeln("<body>") ;
    newWin.document.writeln("<h1>אני הוא העמוד הנוסף</h1>") ;
    newWin.document.writeln("</body>") ;
    newWin.document.writeln("</html>") ;
    buttons.style.visibility = "visible" ;
    button2open.style.visibility = "hidden" ;
   }

   function newWinInfocus()
   {
    if (newWin.closed) {
     alert("Sorry")
    }else{
     newWin.focus() ;
    }
   }

   function closeCurrentWin()
   {
    self.close() ;
   }

   function closeNewWin()
   {
    newWin.close() ;
    button2open.style.visibility = "visible" ;
   }

  </script>
 </head>
<body>

<div id="button2open" style="visibility:visible;">
 <input type="button" value="פתח חלון חדש"
   onclick="makeNewWindow();" />
</div>
<br /><br />

<div id="buttons" style="visibility:hidden;">
 <input type="button" value="עבור לחלון החדש שנפתח"
   onclick="newWinInfocus();" />
 <br /><br />
 <input type="button" value="סגור חלון חדש שנפתח"
   onclick="closeNewWin();" />
 <br /><br />
```

```
</div>
<input type="button" value="סגור את החלון הנוכחי"
  onclick="closeCurrentWin();" />
<br /><br />
</body>
</html>
```

שאלה 10.3

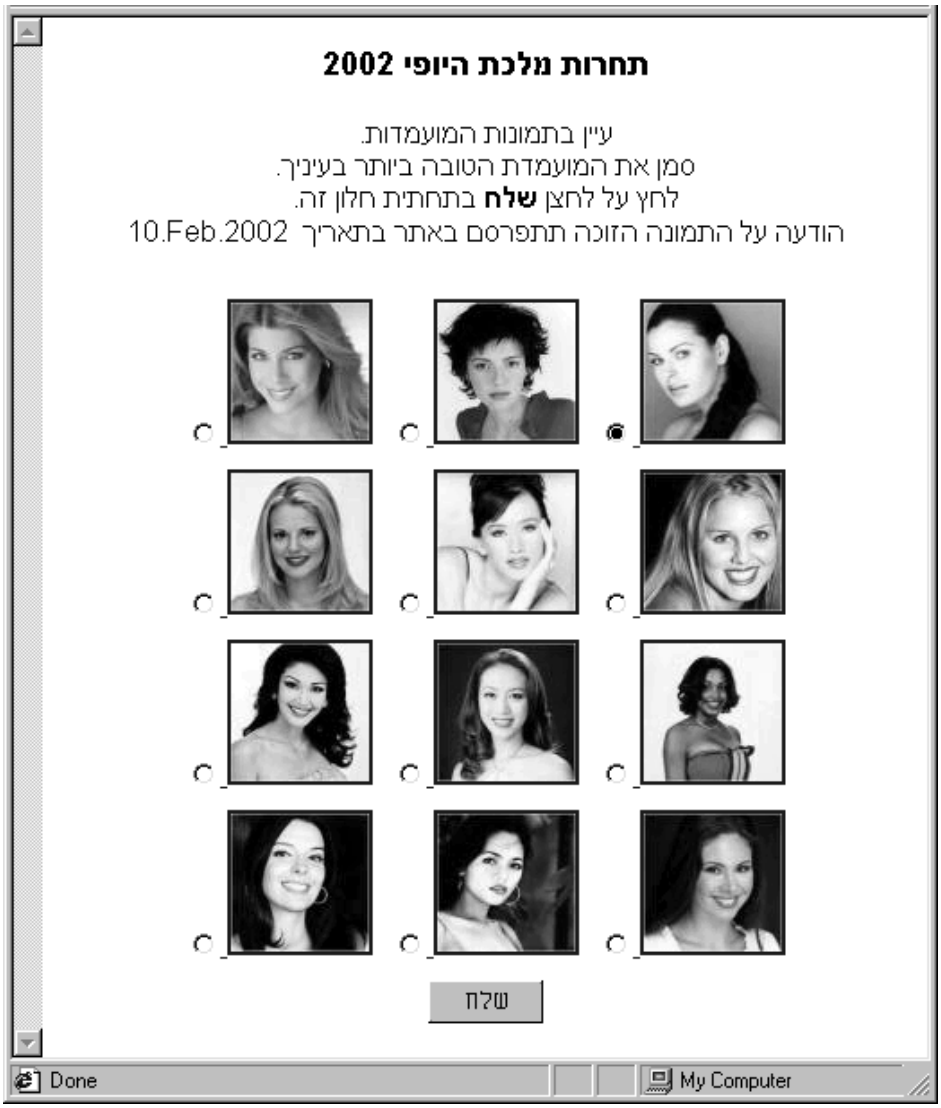
בהמשך לשאלה הקודמת, הוסף שדה טופס מסוג select. הרשימה תכיל 6 צבעים. בחירה בצבע תשנה את צבע הרקע של החלון החדש האחרון שנפתח.

הערה

יהיה עליך לוודא שהחלון החדש אכן נפתח לפני שאתה מנסה לשנות את צבע הרקע שלו.

תחרות מלכת היופי 2002

זוהי דוגמה יפה (כשתפעיל את הקובץ תבין למה אני מתכוון) המשלבת מספר אלמנטים: תמונות (כמובן), טבלה, טופס, פתיחת חלון חדש באמצעות תסריט, מחרוזות והצגת נתוני טופס באמצעות תסריט.



תרשים 10.4: תחרות מלכת היופי.

דרישות היישום:

1. יש להציג תמונות מוקטנות (thumbnails) של 12 מועמדות (התמונות נמצאות באתר בכתובת: www.hod-ami.co.il/59355.html, ויש עוד כמה תמונות בונוס). התמונות יוצגו ב- 4 שורות.
2. כל תמונה מוקטנת תהיה קישור לתמונה גדולה יותר שתיפתח בחלון חדש.
3. החלון החדש שייפתח יהיה ברוחב 280 נקודות ובגובה 302 נקודות.
4. החלון החדש שיציג את התמונה המוגדלת של המועמדת יהיה ללא לחצנים, פסי גלילה וכדומה, רק עם מסגרת סביבו (התמונות נמצאות באתר). החלון ימוקם בפינה השמאלית העליונה של המסך.
5. בחלון החדש שיציג את התמונה המוגדלת של המועמדת יהיה לחצן שלחיצה עליו תסגור את החלון.
6. ניתן לבחור רק מועמדת אחת.
7. יש להציג למשתמש את הבחירה (מספר ושם המדינה אותה מייצגת המועמדת).

להלן רשימת שמות התמונות המוקטנות והמוגדלות של המועמדות (כל התמונות נמצאות באתר בכתובת: www.hod-ami.co.il/59355.html):

תמונה מוקטנת (77x77)	תמונה מוגדלת (220x220)
colombia-s.jpg	colombia.jpg
israel-s.jpg	israel.jpg
peru-s.jpg	peru.jpg
newzealand-s.jpg	newzealand.jpg
india-s.jpg	india.jpg
canada-s.jpg	canada.jpg
jamaica-s.jpg	jamaica.jpg
korea-s.jpg	korea.jpg
venezuela-s.jpg	venezuela.jpg
romania-s.jpg	romania.jpg
philippines-s.jpg	philippines.jpg
bosnia-s.jpg	bosnia.jpg

שים לב למתן השמות העקבי. בשיטה זו נוכל להשתמש בשם התמונה כדי להציג את המדינה אותה מייצגת המועמדת (סעיף 7 בדרישות היישום).

כדי להציג תמונה של מועמדת לפי סעיף 1 בדרישת היישום (למשל, המועמדת של קולומביה), יש לרשום:

```
<img src="images/colombia-s.jpg" />
```

כדי שלחיצה על התמונה תוביל לפתיחת חלון חדש על כל ההגדרות שלו, יש לרשום:

```
<img src="images/colombia-s.jpg"
   onclick="openNewWin('colombia.jpg');" />
```

לפונקציה ()openNewWin שתפתח חלון חדש ותציג את תמונת המועמדת ולחצן, נשלח שם התמונה המוגדלת.

הפונקציה ()openNewWin מקבלת את שם התמונה המוגדלת אותה צריך להציג. כזכור, התמונות נמצאות בתיקיה images ולמעשה היה צריך לשלוח לפונקציה לא רק את שם התמונה אלא גם את הנתיב (path), באופן הבא:

```
<img src="images/colombia-s.jpg"
   onclick="openNewWin('images/colombia.jpg');" />
```

בחרתי להוסיף את הנתיב לשם הקובץ במסגרת הפונקציה ()openNewWin ולא בקריאה לפונקציה ומשלוח פרמטרים, באופן הבא:

```
function openNewWin(imgName) {
     imgName = 'images/' + imgName ;
```

מכיון שהפונקציה ()openNewWin פותחת חלון חדש יש להגדיר את הפרמטרים לפתיחת החלון, לפי סעיפים 3 ו- 4 בדרישות היישום, באופן הבא:

```
myFeatures = 'directories=no,location=no,menubar=no,' ;
myFeatures += 'status=no, titlebar=no, toolbar=no,' ;
myFeatures += 'scrollbars=no,width=280,height=302,' ;
myFeatures += 'resizeable=yes' ;
```

פתיחת החלון החדש והריק לפי הפרמטרים שהוגדרו:

```
nWin = window.open('','pic',myFeatures) ;
```

החלון החדש הוא אובייקט בשם nWin, אליו תוכל להתייחס מתוך תסריט JavaScript. עליך להציב באובייקט את התמונה לפי סעיף 2 בדרישות היישום. אבל איך אפשר לעשות את זה אם החלון ריק ואין בו תוכן? ובכן, יש לבנות מסמך HTML שיכיל את התמונה ואת זה ניתן לעשות בעזרת השיטה ()document.write, באופן הבא:

```
nWin.document.write("<html dir='rtl'>") ;
nWin.document.write("<head><title>Hod-Ami</title></head>");
nWin.document.write("<body>") ;
nWin.document.write("<div align='center'>") ;
nWin.document.write("<br />") ;
nWin.document.write('<img src=') ;
```

```
nWin.document.write("'" + imgName + "'") ;
nWin.document.write(" />") ;
```

למעשה, רק שלוש השורות האחרונות בקוד שלעיל מגדירות את התגית <img />, אך למען שלמות הקוד, יש לבנות את הדף החדש מהתחלה. כלומר, להתחיל בתגית <html>.

ראה את השימוש בשם האובייקט nWin. ללא ציון שם אובייקט החלון, השיטה document.write() היתה כותבת לחלון הנוכחי ממנו היא מופעלת. כדי שיהיה ניתן מחלון אחד לכתוב לחלון שני יש לציין את שמו, ובמקרה שלנו nWin.

סעיף 5 בדרישות היישום דורש הימצאות לחצן שבעזרתו ניתן יהיה לסגור את החלון:

```
nWin.document.write("<input type='button' value='   סגור   '
onclick='window.close();' />") ;
nWin.document.write("</div>") ;
nWin.document.write("</body>") ;
nWin.document.write("</html>") ;
```

סגירת החלון מתבצעת על ידי הפעלת תסריט באירוע onclick של התגית <input />, ולאחריו סוגרים את המסמך באופן מסודר.

לסיום העבודה סוגרים את העברת הנתונים לדף, הופכים את החלון החדש להיות החלון הפעיל ומזיזים אותו לפינה השמאלית העליונה של המסך:

```
nWin.document.close() ;
nWin.focus() ;
nWin.moveTo(0,0) ;
}
```

וזוהי כל הפונקציה לבניית החלון החדש שפתחת ושהציגה בו תמונה ולחצן:

```
function openNewWin(imgName) {

  imgName = 'images/' + imgName ;

  myFeatures = 'directories=no,fullscreen=no,location=no,menubar=no,' ;
  myFeatures += 'status=no,titlebar=no,toolbar=no,' ;
  myFeatures += 'scrollbars=no,width=280,height=302,' ;
  myFeatures += 'resizeable=yes' ;

  nWin = window.open('','pic',myFeatures) ;
  nWin.document.write("<html dir='rtl'>") ;

  nWin.document.write("<head><title>Hod-Ami</title></head>");
  nWin.document.write("<body>") ;
  nWin.document.write("<div align='center'>") ;
  nWin.document.write("<br />") ;
  nWin.document.write("<img src=") ;
```

```
  nWin.document.write("'" + imgName + "'") ;
  nWin.document.write(" />") ;
  nWin.document.write("<br />") ;
  nWin.document.write("<input type='button' value='   סגור   '
   ↳ onclick='window.close();' />") ;
  nWin.document.write("</div>") ;
  nWin.document.write("</body>") ;
  nWin.document.write("</html>") ;
  nWin.document.close() ;
  nWin.focus() ;
  nWin.moveTo(0,0) ;
}
```

להזכירך, התגית להצגת תמונת מועמדת, למשל המועמדת מקולוביה, נכתבה כך:

```
<img src="images/colombia-s.jpg"
   onclick="openNewWin('colombia.jpg');" />
```

למשתמש לא יהיה שמץ של מושג שהתמונה היא קישור. הסמן לא יהפוך ליד ולא תהיה מסגרת כחולה לתמונה. מה עושים? רושמים את התגית <img /> כחלק מתגית <a>, באופן הבא:

```
<a href="javascript:void(0);">
 <img src="images/colombia-s.jpg"
   onclick="openNewWin('colombia.jpg');" />
</a>
```

רישום שכזה מציג את התמונה כקישור: מסגרת בצבע כחול והסמן משתנה ליד.

לפי סעיף 6 של דרישות היישום, ניתן לבחור רק מועמדת אחת. לצורך זה יש לבנות טופס המכיל שדות מסוג Radio. בחרתי להציג את שדה הטופס ליד כל תמונה של מועמדת:

```
<input type="radio" name="picNo" checked="checked" />
```

מכיון שיש להציג את תמונות המועמדות ב- 4 שורות, בחרתי להציג זאת בטבלה. להלן הקוד להצגת 3 המועמדות בשורה הראשונה:

```
<form name="imageForm" action="" method="" onsubmit="check();">
<table border="0" cellpadding="0" cellspacing="10">
  <tr>
   <td valign="bottom" align="center">
    <a href="javascript:void(0);">
     <img src="images/colombia-s.jpg"
       onclick="openNewWin('colombia.jpg');" />
    </a>
```

```
    <input type="radio" name="picNo" checked="checked" />
   </td>

   <td valign="bottom" align="center">
    <a href="javascript:void(0);">
     <img src="images/israel-s.jpg"
       onclick="openNewWin('israel.jpg');" />
    </a>
    <input type="radio" name="picNo" />
   </td>

   <td valign="bottom" align="center">
    <a href="javascript:void(0);">
     <img src="images/peru-s.jpg"
       onclick="openNewWin('peru.jpg');" />
    </a>
    <input type="radio" name="picNo" />
   </td>
  </tr>
```

הסבר לקוד שלעיל:

הגדרת הטופס (<form>) משולבת בהגדרת הטבלה (<table>).

לכל שדות הטופס יש אותו סוג ("type="radio) ואותו שם ("name="picNo). זאת מכיון שהדרישה היא לבחור רק מועמדת אחת.

רק לשדה הטופס הראשון ניתן ערך למאפיין checked, כדי לוודא שהמשתמש לא יוכל לשלוח טופס בלי לבחור בשדה.

באותו אופן מוצגות שורות 2, 3 ו- 4 בטבלת המועמדות.

נותר ליצור את לחצן "שלח טופס" והנה הוא לפניך:

```
<input type="submit" value="   שלח   " />
```

לחיצה עליו תפעיל את ה-URL הנמצא במאפיין action של התגית <form>, אבל אם יש event-handler בשם onsubmit בתגית <form> הוא יבוצע לפני הרשום במאפיין action. לכידת האירוע submit קוראת לפונקציה ()check שתציג את מספר המועמדת ואת המדינה אותה היא מייצגת, לפי סעיף 7 בדרישות היישום.

אתה יודע כמה תמונות יש? בוודאי, יש 12 תמונות. את המספר הזה אפשר להעביר כמספר קבוע לפונקציה ()check או לתת לתוכנית לגלות בעצמה. מכיון שלכל תמונה יש שדה טופס (כחלק מטופס בשם imageForm) ולכל השדות יש אותו שם (picNo), הרי שקיים אוסף (collection) בשם picNo (אוסף הוא מעין מערך) המכיל את כל השדות באותו שם.

מציאת מספר שדות הטופס:

```
function check() {
    noOfItems = document.imageForm.picNo.length ;
```

ב-JavaScript מספרו של האיבר הראשון באוסף הוא אפס (0). מפאת כבוד המועמדת הקולומביאנית, נשדרג את מספרה מ- 0 ל- 1, ואת המועמדת מישראל נשדרג מ- 1 ל- 2. בקיצור, מספר המועמדת יהיה מספרה באוסף (collection) ועוד 1.

כן, אבל איזו מועמדת נבחרה? למציאת התשובה, יש צורך לעבור על כל איברי האוסף picNo ולבדוק את התכונה checked, באופן הבא:

```
for (var i=0 ; i < noOfItems ; i++) {
 tmp = document.imageForm.picNo[i].checked ;
 if (tmp) {
  selItem = i ;
 }
}
```

משתנה selItem יכול לקבל ערכים מ- 0 ועד 11. ערך אחד הוא חייב לקבל, ומדוע? בבניית הטופס בדף HTML ניתן ערך checked למאפיין checked של אחד השדות (למועמדת מקולומביה). כלומר, אם המשתמש לחץ על לחצן **שלח** מבלי לבחור, הרי שלמעשה הוא אישר את הבחירה במועמדת מקולומביה, ולכן במשתנה selItem יוצב הערך 0.

מספר המועמדת לאחר השדרוג יהיה מספרה באוסף בתוספת הערך 1, ואת זה יש להציג כך:

```
candidateNo = selItem + 1 ;
alert("מועמדת מספר: " + candidateNo) ;
```

מהו שם המדינה שמייצגת המועמדת שבחר המשתמש? מכיון שלתמונות המועמדות ניתן שם המדינה: התמונה של המועמדת מקולומביה נקראת colombia-s.jpg, התמונה של המועמדת מישראל נקראת israel-s.jpg, התמונה של המועמדת מפרו נקראת peru-s.jpg וכך הלאה, ניתן יהיה להשתמש בשם התמונה "ולחלץ" את שם המדינה.

התמונות עם סיומת s- מוצגות בטבלת המועמדות והן מוצגות באמצעות התגית <img />. כל התמונות במסמך נמצאות באוסף (collection) בשם images, כך שנוכל לגשת אליהן באמצעות תסריט. בהנחה שרק תמונות המועמדות מוצגות בדף זה, מספר התמונה הראשונה באוסף הוא 0, מספר התמונה השנייה הוא 1 וכך הלאה. זה מזכיר את המספור של שדות הטופס. כלומר, אם המשתמש בחר בתמונה מספר n, אז הוא בחר במועמדת שמספר התמונה שלה הוא n.

```
picPath = document.images[selItem].src ;
```

השם שיתקבל תלוי במיקום הפיסי של התמונה במחשב שלך או בשרת, אבל דבר אחד ידוע לך וזה שהשם הארוך מסתיים ב- /images ולאחריו שם התמונה. למשל, השם של התמונה של המועמדת מקולומביה מסתיים ב:

```
...images/colombia-s.jpg
```

יש לאתר את הסימן / האחרון בעזרת השיטה lastIndexOf(), באופן הבא:

```
lastSlash = picPath.lastIndexOf("/") ;
```

עכשיו צריך "לחלץ" את שם המדינה מהתו שאחרי הסימן / ועד..? אך איך תדע מה אורך המחרוזת שיש "לחלץ", מכיון שמספר התווים של colombia שונה מספר התווים של peru. כלומר, לכל מדינה שם משלה באורך משתנה (לא קבוע בין מדינה למדינה). אבל, אתה יודע ששם התמונה מסתיים במחרוזת s.jpg- או במילים אחרות 6 התווים האחרונים מיותרים:

```
lastChar = picPath.length ;
picName = picPath.substring(lastSlash+1, lastChar-6) ;
```

ולא נותר אלא להציג את המדינה:

```
alert("מדינה: " + picName);
}
```

בזאת הסתיימה הפונקציה check(), והרי היא לפניך:

```
function check() {
    noOfItems = document.imageForm.picNo.length ;
    for (var i=0 ; i < noOfItems ; i++) {
     tmp = document.imageForm.picNo[i].checked ;
     if (tmp) {
      selItem = i ;
     }
    }
    candidateNo = selItem + 1 ;
    alert("מועמדת מספר: " + candidateNo) ;
    picPath = document.images[selItem].src ;
    lastSlash = picPath.lastIndexOf("/") ;
    lastChar = picPath.length ;
    picName = picPath.substring(lastSlash+1, lastChar-6) ;
    alert("מדינה: " + picName);
}
```

הקובץ **formImage.html** לפניך:

```
<html dir="rtl">
 <head>
  <title>
  </title>
  <script language="javascript">
   function check() {
    noOfItems = document.imageForm.picNo.length ;
    for (var i=0 ; i < noOfItems ; i++) {
     tmp = document.imageForm.picNo[i].checked ;
     if (tmp) {
      selItem = i ;
     }
    }
    candidateNo = selItem + 1 ;
    alert("מועמדת מספר: " + candidateNo) ;
    picPath = document.images[selItem].src ;
    lastSlash = picPath.lastIndexOf("/") ;
    lastChar = picPath.length ;
    picName = picPath.substring(lastSlash+1, lastChar-6) ;
    alert("מדינה: " + picName);
   }

   function openNewWin(imgName) {

      imgName = 'images/' + imgName ;

      myFeatures = 'directories=no,fullscreen=no,location=no,menubar=no,' ;
      myFeatures += 'status=no,titlebar=no,toolbar=no,' ;
      myFeatures += 'scrollbars=no,width=280,height=302,' ;
      myFeatures += 'resizeable=yes' ;

      nWin = window.open('','pic',myFeatures) ;
      nWin.document.write("<html dir='rtl'>") ;

      nWin.document.write("<head><title>Hod-Ami</title></head>");
      nWin.document.write("<body>") ;
      nWin.document.write("<div align='center'>") ;
      nWin.document.write("<br />") ;
      nWin.document.write("<img src=") ;
      nWin.document.write("'" + imgName + "'") ;
      nWin.document.write(" />") ;
      nWin.document.write("<br />") ;
      nWin.document.write("<input type='button' value='   סגור   '
      ↳ onclick='window.close();' />") ;
      nWin.document.write("</div>") ;
```

```
      nWin.document.write("</body>") ;
      nWin.document.write("</html>") ;
      nWin.document.close() ;
      nWin.focus() ;
      nWin.moveTo(0,0) ;
   }

  </script>
 </head>
<body>
<div align="center">
 <h3>
תחרות מלכת היופי 2002
 </h3>
עיין בתמונות המועמדות.
<br />
סמן את המועמדת הטובה ביותר בעיניך.
<br />
לחץ על לחצן
<b>
שלח
</b>
בתחתית חלון זה.
<br />
הודעה על התמונה הזוכה תתפרסם באתר בתאריך
<span dir="ltr">
10.Feb.2002
</span>

<br />
<form name="imageForm" action="" method="" onsubmit="check();">
 <table border="0" cellpadding="0" cellspacing="10">
  <tr>
   <td valign="bottom" align="center">
    <a href="javascript:void(0);">
     <img src="images/colombia-s.jpg"
       onclick="openNewWin('colombia.jpg');" />
    </a>
    <input type="radio" name="picNo" checked="checked" />
   </td>
```

```
   <td valign="bottom" align="center">
    <a href="javascript:void(0);">
     <img src="images/israel-s.jpg"
       onclick="openNewWin('israel.jpg');" />
    </a>
    <input type="radio" name="picNo" />
   </td>

   <td valign="bottom" align="center">
    <a href="javascript:void(0);">
     <img src="images/peru-s.jpg"
       onclick="openNewWin('peru.jpg');" />
    </a>
    <input type="radio" name="picNo" />
   </td>
  </tr>

<!-- line 2 -->
  <tr>
   <td valign="bottom" align="center">
    <a href="javascript:void(0);">
     <img src="images/newzealand-s.jpg"
       onclick="openNewWin('newzealand.jpg');" />
    </a>
    <input type="radio" name="picNo" />
   </td>

   <td valign="bottom" align="center">
    <a href="javascript:void(0);">
     <img src="images/india-s.jpg"
       onclick="openNewWin('india.jpg');" />
    </a>
    <input type="radio" name="picNo" />
   </td>

   <td valign="bottom" align="center">
    <a href="javascript:void(0);">
     <img src="images/canada-s.jpg"
       onclick="openNewWin('canada.jpg');" />
    </a>
    <input type="radio" name="picNo" />
  </tr>
```

```
<!-- LINE 3 -->
  <tr>
   <td valign="bottom" align="center">
    <a href="javascript:void(0);">
     <img src="images/jamaica-s.jpg"
       onclick="openNewWin('jamaica.jpg');" />
    </a>
    <input type="radio" name="picNo" />
   </td>

   <td valign="bottom" align="center">
    <a href="javascript:void(0);">
     <img src="images/korea-s.jpg"
       onclick="openNewWin('korea.jpg');" />
    </a>
    <input type="radio" name="picNo" />
   </td>

   <td valign="bottom" align="center">
    <a href="javascript:void(0);">
     <img src="images/venezuela-s.jpg"
       onclick="openNewWin('venezuela.jpg');" />
    </a>
    <input type="radio" name="picNo" />
   </td>
  </tr>

<!-- line 4 -->

  <tr>
   <td valign="bottom" align="center">
    <a href="javascript:void(0);">
     <img src="images/romania-s.jpg"
       onclick="openNewWin('romania.jpg');" />
    </a>
    <input type="radio" name="picNo" />
   </td>

   <td valign="bottom" align="center">
    <a href="javascript:void(0);">
     <img src="images/philippines-s.jpg"
       onclick="openNewWin('philippines.jpg');" />
    </a>
    <input type="radio" name="picNo" />
   </td>
```

```
   <td valign="bottom" align="center">
    <a href="javascript:void(0);">
     <img src="images/bosnia-s.jpg"
       onclick="openNewWin('bosnia.jpg');" />
    </a>
    <input type="radio" name="picNo" />
   </td>
  </tr>

 </table>

  <input type="submit" value="   שלח   " />

</form>

</div>
</body>
</html>
```

בזאת סיימנו לבנות את דף מלכות היופי. אל "תיתקע" מול דף זה, יש עוד דברים אחרים לעשות, למשל... להפוך את הדף ולהתקדם בפרויקט סיינפלד.

פרויקט סיינפלד

פתיחה של דף בחלון חדש אינו משהו יוצא דופן והוא מתבצע באמצעות המאפיין target של התגית <a>. שליטה דרך תסריט על פתיחת החלון יכולה לתת כמה אפשרויות נוספות כמו: סגירה של החלון החדש, שינוי פרמטרים שלו כמו צבע, גודל וכדומה ועוד. למעשה, כל מה שניתן לעשות בחלון הפעיל אפשר גם לבצע על חלון חדש שנפתח על ידי תסריט.

בדף about.html נמצא הקישור הבא:

```
<a href="images\BabuBig.jpg"
       title="תמונה מוגדלת של היועץ" target="_blank">
  בבו
</a>
```

לחיצה על הטקסט **בבו** תוביל לפתיחת חלון חדש בו תוצג התמונה BabuBig.jpg. היתרון בקישור זה שהוא פשוט וקל לכתיבה, אבל האם יש לך שליטה על גודל החלון הנפתח? האם אתה קובע היכן על המסך ייפתח החלון? התשובה היא - לא.

נכתוב תסריט שיפתח חלון חדש ויציג בו תמונה:

```
<a href="JavaScript:void(0);"
 onclick="window.open('images/BabuBig.jpg');"
 title="תמונה מוגדלת של היועץ">
  בבו
</a>
```

הפעלת הקישור תציג לנו חלון חדש של IE בגודל, במיקום שאין לה שליטה עליו.

נוסיף תכונות לחלון החדש:

```
<a href="JavaScript:void(0);"
onclick="window.open('images/BabuBig.jpg','babu','width=263,
↳height=533, toolbar=no, resizable=yes, scrollbars=no,
↳menubar=no, status=no, location=no, top=0, left=0');"
 title="תמונה מוגדלת של היועץ">
  בבו
</a>
```

בכוונה רשמתי את כל תכונות החלון מכיון שאני לא מצפה ממך, ומאף תוכניתן, לזכור בעל פה את כל התכונות ואת ברירת המחדל של כל תכונה ותכונה. סיבה נוספת, כל שינוי שתרצה לערוך תוכל לעשות מייד בלי צורך לגשת למקורות (הכוונה היא לספרי מחשבים).

הסיבה שבחרתי לקבוע את רוחב החלון ל-263 נקודות ואת גובהו ל-533 נקודות היא שאלו הן מידות התמונה BabuBig.jpg, אבל כאמור יכולתי לקבוע כל גודל אחר. זכור שגודל התמונה לא תותאם לגדול החלון. מכיון שיש שוליים, עדיף לקבוע את גודל החלון מעט יותר גדול מגודל התמונה.

נעשה אותו דבר עם התמונה של קייט הרנאנדוס. המידות של התמונה KeithHernandez.jpg הם : 297x384, בעוד שמידות החלון נקבעו ל- 320x420.

אפשר לגשת לפתיחת החלון בגישה כללית יותר ולספק פונקציה שתפתח חלון לפי ערכי הפרמטרים שהועברו אליה.

סיכום

בחלק זה, החלק השני של הספר **מבוא לתכנות בסביבת אינטרנט, JavaScript**, הונחה התשתית לבניית אתר אינטרנט אינטראקטיבי. בניית תסריטים שהגיבו לפעולת המשתמש בעת הזזת העכבר. לחיצה על אלמנט (ולא דווקא קישור). בניית תסריטים הבודקים את הטופס שהקליד המשתמש בטפסי HTML ועוד. בחלק זה לקחת שליטה.

הרחבות לחומר הלימוד:

נושא	ספר בהוצאת **הוד-עמי**
יסודות התכנות עם JavaScript בסביבת אינטרנט	**JavaScript סדנת לימוד**
מודל האובייקטים JavaScript + CSS + DOM Dynamic HTML	**JavaScript+CSS+DOM למפתחי אתרים באינטרנט**
פירוט מקיף של כל האובייקטים, האוספים, השיטות והתכונות	**JavaScript המדריך השלם**
עיצוב האתר	**עיצוב ממשק באינטרנט**

הדפים אותם בנית בעזרת HTML הם דפים סטטיים, עכשיו הוספת להם יכולת אינטראקטיבית באמצעות תסריטים הכתובים JavaScript. אז מה עושים עם נתוני הטופס? בסדר, ביצעת בנתוני הטופס בדיקות תקינות ומצאת שהם נרשמו כנדרש, ומה הלאה?

השלב הבא הוא לשמור את הנתונים שבטופס (ולא רק) בשרת. למשל, לרשום בשרת את בחירת המועמדת למלכת היופי שבחרת. בחירתך תצטרף לבחירות של גולשים אחרים. למשל, בדיקת נתוני ההרשמה שלך למועדון לקוחות, לא רק ברמה המקומית: האם רשמת דואר אלקטרוני חוקי, אלא בדיקה האם אתה רשום כבר במועדון הלקוחות ואת זה ניתן לבצע... בשרת.

שילוב יכולת עבודה מול שרת תאפשר אינטראקטיביות ברמה גבוהה עוד יותר בין האתר למשתמש בו. זוהי טכנולוגיית **ASP**. המשמעות היא שחלק מהדף שלך יפעל בשרת וחלקו במחשב המקומי. כלומר, תוכל לכתוב תוכניות שלמות שיתבצעו בשרת בלבד ועוד. כדי להפעיל את טכנולוגיית ASP אינך זקוק לשום תוספת מעבר למה שיש לך עכשיו במחשב הלקוח שלך. יהיה עליך לדאוג שמחשב השרת יכול לתמוך ב-ASP ואת זה מאפשרים השרתים של Microsoft. את כל זה תלמד בחלק הבא:

מבוא לתכנות בסביבת אינטרנט, ASP

אינדקס

האינדקס מופיע בעברית ובאנגלית.

כיוון הקריאה של האינדקס באנגלית הוא מסוף הספר.

כלומר, פתח את הספר בסופו ושם תמצא את תחילת האינדקס באנגלית - עמוד 1.

אינדקס

א

אובייקט, 15-20
HTML ו-, 18
this, 99
אוספים, 19-20
בתוך אובייקט, 19
מכולה, 15
שיטות, 16-17
שם, 95-96
תחביר נקודה, 15
תכונות, 16
אובייקט Global, 131
אובייקט בתוך אובייקט, 19
אובייקט החלון, 181-207
אובייקט המסמך, 35
אובייקט מחרוזת, 127-160
אובייקט תאריך, 161-169
אובייקט תמונה, 93-125
אוספים, 19-20
אופרטורים
מתמטיים, 46
השוואה, 47
השמה, 47
מקוצרים, 48, 56-57
בדיקת תנאי, 47
לוגיים, 47
סימן שוויון, 47
מקוצרים (השמה), 48, 56-57
אורך מחרוזת, 128
אירועים, 77-80, 94-95
אנימציה, 108-112

ב

בדיקת נתוני טופס, 128-130
בדיקת תנאי, 47
בוליאני (ערך משתנה), 44

ג

ג'אווה-סקריפט
אופרטורים
מתמטיים, 46
השוואה, 47
השמה, 47
מקוצרים, 48, 56-57
בדיקת תנאי, 47
לוגיים, 47
סימן שוויון, 47
בהשוואה לשפת C, 25-26
הסרת קוד JavaScript, 27
משתנים, 26
גלובלי, 108
הגדרה, 42
הצהרה, 42
חופשיים, 43
מקומי, 108
ערך
בוליאני, 44
לא-מוגדר, 44
מחרוזת, 43
מספר, 44

שם, 39-40
השיטה ההונגרית, 40
מילות מפתח, 39
כתיבת JavaScript, 27
מילות מפתח, 39
רגישות לגודל אות, 31
גלובלי (משתנה), 108

ה

הגדרת מערך, 102-105
הגדרת משתנה, 42
הידור, 26
החזרת ערך מפונקציה, 86-88
הסרת קוד JavaScript, 27
העברת מידע בין חלונות, 185-187
העברת נתונים לפונקציה, 97-99
הצבה (מערך חד-מימדי), 102-105
הצהרת משתנה, 42
השוואה (אופרטורים), 47
השיטה ההונגרית, 40
השמה (אופרטורים), 47
מקוצרים, 48, 56-57
השמה (מערך חד-מימדי), 102-105

ח

חד-מימדי (מערך), 102-112
הגדרה, 102-105
הצבה, 102-105
השמה, 102-105
שיטות, 106-107
חלון, 181-207
העברת מידע בין, 185-187
פתיחת חדש, 181
חלון הודעה, 28

ט

טופס, 128-133
בדיקת נתוני, 128-130
רשימה נגללת (Select), 143-149
שדה אפשרויות (Radio), 149-153
שינוי ערך (Text), 133-134
תיבת סימון (CheckBox), 157
טעינה מוקדמת (תמונה), 100-101

י

יתרונות השימוש במערך, 105-106

ל

לא-מוגדר (ערך משתנה), 44
לולאות
break, 68
continue, 67
for, 58-59
while, 53-57
מקוננות, 59-62
לקוח (מחשב), 13

מ

מאפיין, 27
מבוסס אובייקטים, 14
מודל האובייקטים, 20, 95
מחרוזת
אורך, 128
מספר תווים, 128
מחרוזת (ערך משתנה), 43
מחשב
לקוח, 13
שרת, 13
מילות מפתח, 39
מיקום פונקציה בתוכנית, 81
מוכוון אובייקטים, 14

מכולה, 15
מערך
חד-מימדי, 102-112
הגדרה, 102-105
הצבה, 102-105
השמה, 102-105
שיטות, 106-107
יתרונות שימוש, 105-106
רב-מימדי, 112-118
הגדרה, 113
מסגרות, 171-180
דינמית, 172
מועדפים, 178
סטטית, 172
תגיות, 171
תזמון, 176
מספר (ערך משתנה), 44
מספר תווים במחרוזת, 128
מקומי (משתנה), 108
משתנים, 26
גלובלי, 108
הגדרה, 42
הצהרה, 42
חופשיים, 43
מקומי, 108
ערך
בוליאני, 44
לא-מוגדר, 44
מחרוזת, 43
מספר, 44
שם, 39-40
השיטה ההונגרית, 40
מילות מפתח, 39
מתירנית (שפת תכנות), 26

ס

סיינפלד, ראה פרויקט סיינפלד
סימן שוויון, 47

ע

ערך משתנה
בוליאני, 44
לא-מוגדר, 44
מחרוזת, 43
מספר, 44

פ

פונקציות, 81-88
החזרת ערך מ, 86-88
העברת נתונים ל, 97-99
מיקום בתוכנית, 81
קריאה ל, 82-84
פרויקט סיינפלד
פרק 5, 89
פרק 6, 119
פרק 7, 154
פרק 8, 169
פרק 9, 179
פרק 10, 206
פתיחת חלון חדש, 181

ק

קינון לולאות, 59-62
קריאה לפונקציה, 82-84

ר

רגישות לגודל אות, 31
רשימה נגללת (Select), 143-149

ש

שדה אפשרויות (Radio), 149-153
שיטות (אובייקט), 16-17
שם
אובייקט, 95-96
משתנה, 39-40
השיטה ההונגרית, 40
מילות מפתח, 39
שינוי ערך (Text), 133-134
שעון, 161-169
שפת תכנות
הידור, 26
מבוסס אובייקטים, 14
מוכוון אובייקטים, 14
מתירנית, 26
שרת (מחשב), 13

ת

תאריך, 161-169
תגית
כתיבת JavaScript, 27
שם, 95-96
תזמון בין מסגרות, 176
תחביר נקודה, 15
תיבת סימון (CheckBox), 157
תכונות אובייקט, 16
תמונה
אובייקט, 93
החלפה, 100
טעינה מוקדמת, 100-101
תגית, 93
תנאים
if, 62
if...else, 62-64
switch, 70
תסריט, 14

V

Vlaues (Variable)
    Boolean, 44
    Number, 44
    String, 43
    Undefined, 44
Var, 42
Variables, 39-57
    Declare, 42
    Free, 43
    Global, 108, 141
    Local, 108
    Name, 39-40
        Hungarian Convention, 40
    Vlaues
        Boolean, 44
        Number, 44
        String, 43
        Undefined, 44
void, 82-83

W

while (loop), 53-57
Windows, 181-207
    open new, 181-182
    passing information between, 185-187
window (object)
    Methods
        alert, 28
        clearInterval, 110, 140
        open, 181-182
        setInterval, 110-112, 140
    Properties
        closed, 189
        status, 163
write (document, object), 37

sort (array, method), 106
split (String, Method), 137
Static Frame, 172-173
String (object)
    Methods
        charAt, 137
        indexOf, 134
        lastIndexOf, 136, 200
        split, 137
        substring, 137
        toLowerCase, 142
        toUpperCase, 142
    Properties
        length, 127-128, 130
String (variable), 43
substring (String, Method), 137
switch, 70

T

Tags
    <!-- -->, 27
    <frame />, 171
    <frameset>, 171
    <img />, 93
    <script>, 27
Text (object), 128-134
Timing (Frame), 176
toLowerCase (String, Method), 142
toUpperCase (String, Method), 142

U

Undefined (variable), 44

Programming
    Client side, 14
    Compilation, 26
    JavaScript, 14
    Object Based, 14
    Object Oriented, 14
Properties
    id, 95
    Language, 27
    name, 95
    Object, 16

R

Radio (Form, Field), 149-153, 197
Radio (object)
    Properties
        checked, 149-150, 197
        length, 150
Return Value from Function, 86-88
reverse (array, method), 106

S

Script
    JavaScript, 14
Select (Form, Field), 143-149
Select (object)
    Properties
        length, 143
        selectedIndex, 143-144
    Collection
        options, 143-145
Seinfeld Project
    Chapter 5, 89
    Chapter 6, 119
    Chapter 7, 154
    Chapter 8, 169
    Chapter 9, 179
    Chapter 10, 206
Shorthand Operators, 48, 56-57

Object
Date, 161-169
Form, 129
Global, 131
Image, 93-125
Radio, 149-152, 197
Select, 143-149
String, 127-160
Text, 128-134
Windows, 181-207
Object Based Programming, 14
Object Oriented Programming, 14
One-dimension Array, 102-112
Assignment, 102-105
defining, 102-105
Methods, see Array (object)
Operators
Arithmetic, 46
Assignment, 47
Shorthand, 48, 56-57
Comaparison, 47
Conditional, 47
Logical, 47

P

Parameters (Function), 97-99
parent (object), 172
parseFloat (Global, Method), 131
parseInt (Global, Method), 131
Passing Value(s) to Function, 97-99
Picture
Changing, 100
Pre-Load, 100-101
Tag, 93
Place a Function in HTML file, 81
Place a Function in Script, 81
Pre-Load (images), 100-101

L

lastIndexOf (String, Method), 136, 200
Local Variable, 108
Logical Operators, 47
Loops
    break, 68
    continue, 67
    for, 58-59
    Nesting, 59-62
    while, 53-57
Loosely Type Language, 26

M

Methods, 16-17
Multi-dimension Array, 112-118
    Assignment, 113-114
    defining, 113

N

Name
    Frame, 172
    Object, 95-96
    Property, 95
Nesting (loops), 59-62
Number (variable), 44

O

Object Oriented Programming, see OOP
Object (general), 15-20
    Collections, 19-20
    Container, 15
    Dot syntax, 15
    HTML and, 18
    Methods, 16-17
    Name, 95-96
    Properties, 16

J

JavaScript
- Case Sensitive, 31
- Keywords, 39
- Loosely Type Language, 26
- Operators
  - Arithmetic, 46
  - Assignment, 47
    - Shorthand, 48, 56-57
  - Comaparison, 47
  - Conditional, 47
  - Logical, 47
- Variables, 39-57
  - Declare, 42
  - Free, 43
  - Global, 108, 141
  - Local, 108
  - Name, 39-40
    - Hungarian Convention, 40
  - Vlaues
    - Boolean, 44
    - Number, 44
    - String, 43
    - Undefined, 44
- vs. C, 25-26

JavaScript:, 82
JavaScript:void(0), 82-83, 186
join (array, method), 106

K

Keywords, 39

G

getDate (date, method), 162
getHours (date, method), 162
getMinutes (date, method), 162
getSeconds (date, method), 162
getYear (date, method), 162
Global (object)
    Methods
        isNaN, 132
        parseFloat, 131
        parseInt, 131
Global Variable, 108, 141

H

Hungarian Naming Convention, 40
HTML and Objects, 18
HTTP, 13

I

id (Tag, attribute), 95
if, 62
if...else, 62-64
Images (collection), 199
Image (object)
    Properties
        src, 94-95
indexOf (String, Method), 134
isNaN (Global, Method), 132

Collections
images, 199
Document Object Model, see DOM
DOM, 20, 95, 98, 129
Dot syntax (object), 15
Dynamic Frame, 172, 174

E

Events, 77-80, 94
Event-Handlers, see Events

F

Favorites (Frames), 178
for (loop), 58-59
Form (object), 129
Form, 128-133
Changing Text Field, 133-134
Checkbox, 157
Checking, 128-130
Radio, 149-153, 197
Select, 143-149
Frames, 171-180
Dynamic, 172, 174
Favorites, 178
Name, 172
parent (object), 172
Static, 172-173
Tags, 171
timing, 176
Free Variable, 43
Functions
Calling, 82-84
Parameters, 97-99
Passing Value(s), 97-99
place in HTML file, 81
place in Script, 81
Return Value, 86-88

C

Calling (Function), 82-84
Case Sensitive Language, 31
Changing Text Field (Form, Field), 133-134
Changing Pictures, 100
charAt (String, Method), 137
Checkbox (Form, Field), 157
CheckBox (object)
    Properties
        checked, 157
Checking (Form, Field), 128-130
Client side programming, 14
Clock, 161-169
Collections, 19-20
Compilation, 26
Container, 15
Comaparison Operators, 47
Conditional Operators, 47
continue, 67
Compiler, 26
Conditional
    if, 62
    if...else, 62-64

D

Date (object)
    Methods
        getDate, 162
        getHours, 162
        getMinutes, 162
        getSeconds, 162
        getYear, 162
        setTime, 167
Declare a Variable, 42
document (object)
    Properties
        bgColor, 36, 181
    Methods
        write, 37

Index

/* */, 32
//, 32
;, 31
`, 31
", 31

A

Animation, 108-111
Arithmetic Operators, 46
Assignment Operators, 47
    Shorthand, 48, 56-57
Array (object)
    Methods, 106-107
        join, 106
        reverse, 106
        sort, 106
    Properties
        length, 105
Array
    One-dimension, 102-112
        Assignment, 102-105
        defining, 102-105
        Methods, see Array (object)
    Multi-dimension, 112-118
        Assignment, 113-114
        defining, 113

B

bgColor (document, property), 36, 181
Boolean, 44
break, 68