HTML5

© אלישבע קצנלנבוגן

HTML5

רקע:

HTML5 מכיל כ- 100 ספסיפיקציות הקשורות לדור הבא של טכנולוגיות ה- Web. השם HTML5 מאגד אותם ביחד. בפועל, HTML5 מאגד חידושים ב- HTML, ב- CSS3 וב- JavaScript שנועדו לאפשר למפתחים לבנות את הדור הבא של אפליקציות ה- Web.

היסטוריה:

- HTML (Hyper Text Markup Languag) נוצר בשנת 1991 כדרך להעברת טקסט, תמונות וקישורים לדפים אחרים ברשת.
- בשנת 1994-1995 העולם התקדם לתקן HTML2 שהכיל תכונות חשובות כמו טבלאות, העלאת קבצים וכדמה.
- בשנת 1996 התקבלו המושגים CSS1 (גליון סגנון) ו - JavaScript (קטעי קוד לדפי html) שקידם את יכולות ה - HTML מדף המסוגל להציג מידע סטטי בלבד לדף עם מידע דינמי .
- בשנת 1997 התקדם התקן ל - HTML4 עם יכולות ה - Frames ואחרים.
- בשנת 1998 - התקדם ה - CSS לגרסה 2, עם תכונות חדשות לעיצוב.
- בשנת 2000 - השתנה התקן ל - XHTML וגרם למפתחי ה - HTML לכתוב קוד יותר מוקשח ותקני.
- בשנת 2005 נכנס לתקן המושג AJAX (גישה א-סינכרונית מהלקוח לשרת).
- החל משנת 2009 התחילו לדבר על HTML5.

HTML5:

מה הן הסיבות שבעטיים נרצה לעבור ל - HTML5?

- התקן הנוכחי אינו מתאים לאתרים מודרניים ואינו מכיל מספיק תכונות לכתיבה נוחה.
- עימוד האתר בעזרת Divs או Tables אינו נכון לוגית, מכיוון שאין משמעות לאלמנטים (div, tr) בהקשר של התוכן שהוא מכיל. אנו מצפים להשתמש בתגים בעלי משמעות לתוכן שהם מכילים (nav עבור תוכן שמכיל לינקים וכדומה)
- הצורך לכתוב קוד שונה עבור כל דפדפן - בעוד שמבחינתנו הדפדפן הוא האמצעי (אירוח האפליקציה) ולא המטרה.

התקן:

התקן ייסגר ככל הנראה בשנת 2022. לא כל הדפדפנים תומכים בכל התכונות, ולא כולם מממשים אותם בצורה זהה בשלב זה.

מה כולל HTML5:

HTML – תגיות חדשות

CSS3 - מאפשר מאפיינים חדשים כמו border-raduis, css selctors.

JavaScript API – גרירת פקדים על המסך, יכולת הרצה מקבילית של קוד, זניחת הכתיבה של Cookies, כתיבה לבסיס נתונים בצד הלקוח, ועוד.

מתחילים...

איך מתחילים?

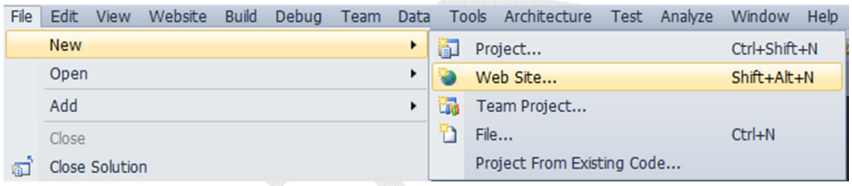
דף html5 יוצרים בדיוק כמו דף html. ניתן לערוך את הדף בכל עורך טקסט. visual studio מציע עורך טקסט ידידותי, ולכן נשתמש בו.

השלבים ליצירת דפי html5:

א. יצירת אתר

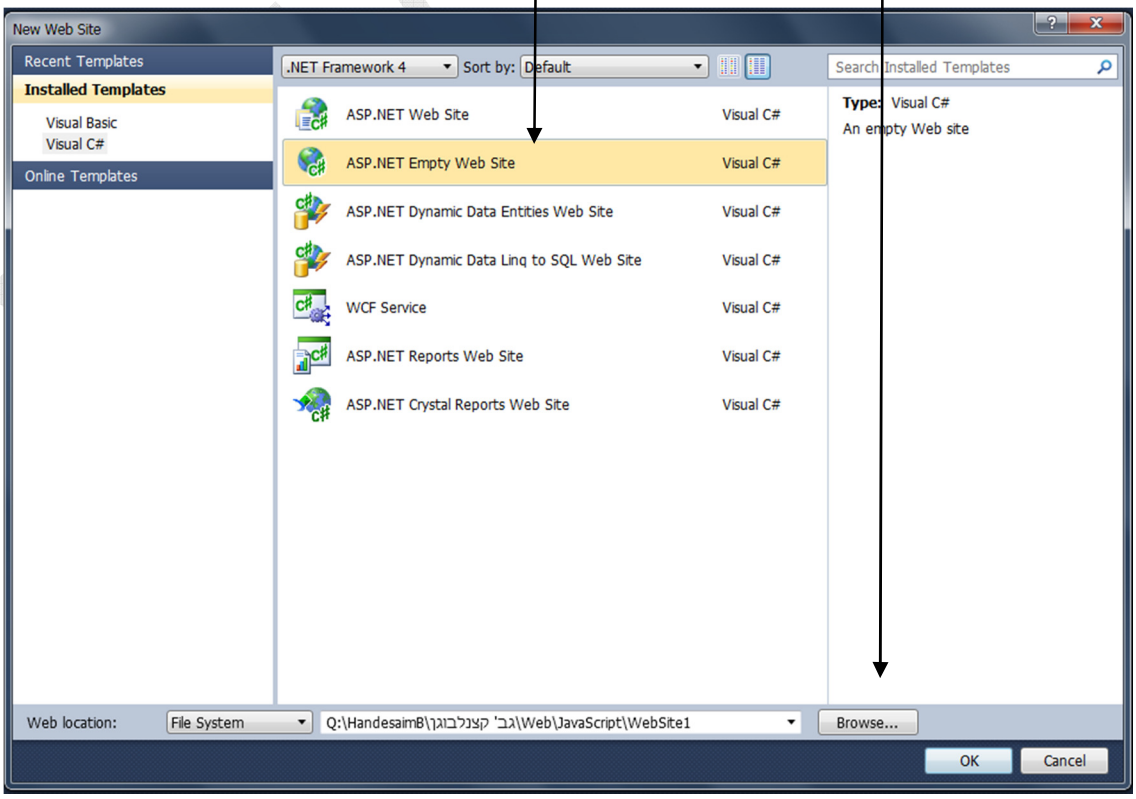
יש לפתוח את ה visual studio, ולייצר web site חדש.

file -> new -> website



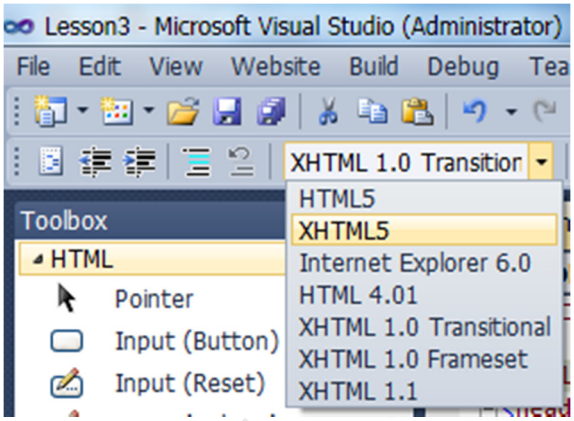
בחלון הנפתח יש לבחור פרויקט מסוג **ASP.NET Empty Web Site**

בלחצן Brows ממקמים היכן האתר ישמר.



ב. הגדרת השימוש ב html5

בסרגל הכלים בראש הדף יש לבחור את סוג הקידוד הנתמך בדף.



אם הסרגל אינו מופיע, ניתן להגדיר זאת גם באופן הבא:

Tools - > options -> TextEditor -> HTML -> validation -> html5

פרק 1 – חידושים ב HTML

ראשי פרקים:

- תגיות סמנטיות
- תגיות טקסט
- תגיות מולטימדיה
- תגיות נוספות

מבנה הדף

מבנה העמוד וה-Doctype ב-HTML 5 שונים לחלוטין מדפים בתקינה קודמת.

אחד הדברים שמאפיינים יותר מכל מסמך של HTML 5 הוא ה-doctype שלו שהוא פשוט לחלוטין. כך נראה doctype תקני של HTML 5:

```
<!DOCTYPE html>
```

שינויים נוספים:

- תגית המטא של charset, שאמורה להגדיר את קידוד הדף, כוללת רק את סוג הקידוד.
- תגית link, אשר כוללת הפניה למשאבים חיצוניים (בדרך כלל css) לא צריכה להכיל את ה-type. אפשר להסתפק רק ב-href וב-rel.
- תגית הסקריפט גם כן משתנה. אין חובה לציין את ה-type בתגית. כמובן שהסקריפט יחשב לסקריפט מבוסס JavaScript אם אין שום ציון של type.

תגיות סמנטיות

כל התגיות בקבוצה זו, אינן משפיעות על המראה העיצובי, אלא נותנות משמעות סמנטית לחלקי המסמך (האתר).

מהי סמנטיקה?

סמנטיקה היא חקר המשמעות - מושג המתאר הבנת המשמעות של מילים בכל הרבדים שלהם.

לדוגמא: תוכנה או מנוע חיפוש שמבינים את ההבדל בין המונחים: מכירה מלשון sale לבין מכירה מלשון know.

כך למשל, מנוע חיפוש סמנטי הוא מנוע חיפוש שכאשר מחפשים בו את המונח ר"ח הוא יבין שמדובר ראשי תיבות של "ראש חודש" ויחזיר תוצאות בהתאם.

התפתחות מנועי החיפוש הסמנטיים ויכולות סמנטיות משופרות בגוגל עצמו בצד מיזמים סמנטיים נוספים הופכים את הסמנטיקה ואת ה-HTML הסמנטי לרלוונטיים יותר ויותר.

מהי תגית סמנטית, ולמה?

בצד היכולות החדשות והמשופרות של HTML 5, הוא מכיל גם תגיות סמנטיות. תגיות סמנטיות הן תגיות שלאו דווקא משפיעות על אופן הצגת התוכן אלא משפיעות הרבה יותר על אפשרות להבנת התוכן וההקשר של התוכן על ידי **מכונה**.

עד היום ניתן היה לכתוב כל תוכן טקסטואלי באחד מתגיות הטקסט: p, span וכדומה.

לדוגמא:

 5 הורדים 'רח

אדם שקורא זאת מבין שמדובר בכתובת, אך מכונה, כמו מנוע החיפוש של גוגל, לא תדע לזהות זאת.

לעומת זאת, תגית חדשה בתקן html5:

<address> 5 הורדים 'רח </address>

תגית כזו גם מכונה יכולה לזהות ולהבין את התוכן וההקשר.

בנייה סמנטית היא חשובה לא רק עבור גוגל, בעתיד דפדפנים חכמים יוכלו לעשות שימוש נוסף בתגיות הסמנטיות על מנת לספק מידע לאוכלוסיות נגישות.
עבור מתכנתים, השימוש בתגיות סמנטיות יכול להקל על הצבת תוספות בדף. למשל, ניתן לשתול JavaScript שלוקח את הכתובת, קורא למפה כלשהי ומציג את הכתובת על גבי מפה. גם היכולות של הסמנטיקה לסייע ב -SEO (קידום מנועי חיפוש)מכריעה את הכף בעד שימוש בסמנטיקה.

רשימת התגיות:

<header>
תחילת העמוד – תגית זו תוחמת בתוכה את החלק העליון במסמך האתר, החלק שתואם לכל שאר הדפים. תגית זו יוצרת למעשה הפרדה של החלק העליון שברוב הפעמים הוא לא החלק המרכזי של הדף.
<footer>
כמו התגית הקודמת, אך הפעם התיחום הוא של החלק התחתון שכולל בדרך כלל מידע משפטי וזכויות יוצרים, קישורים למפת אתר ולחלקים אחרים וכתובות.
<aside>
תגית התוחמת בתוכה את החלק באתר שנמצא בצד. מדובר בתפריט הקבוע שמכיל גם אלמנטים של ניווט (שעבורם יש תגית נוספת) וגם באנרים ומידע קבוע.

<nav>

תגית זו תוחמת בתוכה את החלק במסמך אשר דרכו ניתן לנווט באתר. לרוב, התפריט של האתר.

<section>

תגית זו מחלקת את המסמך לחלקים, תגית זו היא כללית ונשתמש בה כאשר לא נמצא תגית מתאימה יותר. חלופה לשימוש (המוגזם) בתגית div.

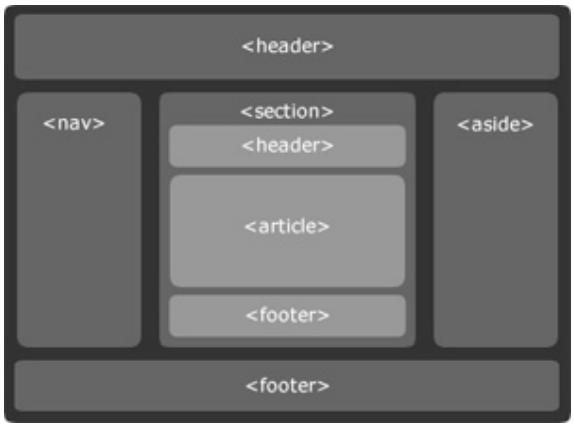
<article>

תגית שמסמנת את המידע באתר בה"א הידיעה – המיקום המרכזי של התוכן שיש בדף: בין אם זה מאמר, תמונה או כל אלמנט מידע אחר שמהווה את מרכז החשיבות של הדף.

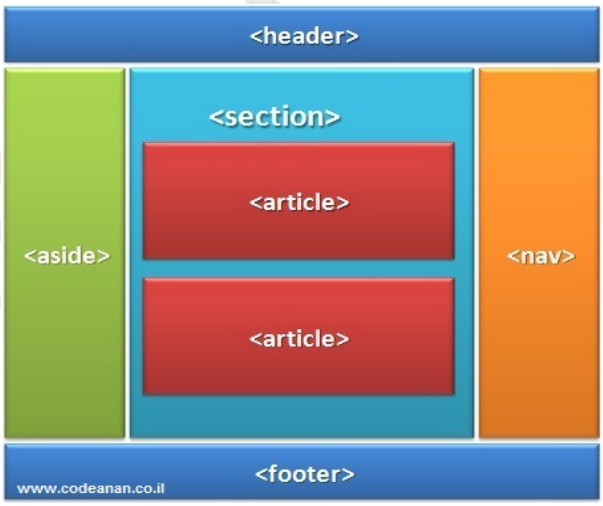
דוגמאות:

איור מס' 1.

איור מס' 2.



איור מס' 3.



<u>שימי לב:</u> תגיות header ו footer אינן חייבות להופיע פעם אחת בכל דף. ניתן לשלב אותן בכל מקום בעל משמעות מתאימה, בדומה לתגיות thead ו tfoot של טבלאות. (ראי איור מס' 2).

דוגמא להחלפת ה- div בתגיות הסמנטיות:

HTML5 REDUCES THE NEED FOR ID AND CLASSES

HTML 4

```
<div id="header">
<h1>Welcome to our site</h1>
</div>
<div id="mainContent">
<h2>The main content</h2>
<p>intro paragraph</p>
<p>The body copy</p>
</div>
<div id="footer">
<p>footer content</p>
</div>
```

HTML5

```
<header>
<h1>Welcome to our site</h1>
<header>
<section>
<header>
<h2>The main content</h2>
<p>intro paragraph</p>
</header>
<p>The body copy</p>
</section>
<footer>
<p>footer content</p>
<footer>
```

סידור אלמנטים בדף HTML

נית וסדר אלמנטים בדף HTML ב- 2 אופנים: טבלאות ו div.

באמצעות טבלה מחלקים את הדף לחלקים הרצויים, ובם ממקמים את התכנים המתאימים.

באמצעות div ממקמים את האלמנטים בתוך divים אותם מסדרים באמצעות ציפת אלמנטים ע"י תכונות של שפת css.

ב- html5 סידור האלמנטים בטבלאות הינה גישה לא מקובלת, לכן מובא פרק זה המלמד כיצד ניתן לסדר אלמנטים בדף באמצעות css, על אף שהדברים תקפים גם לגרסאות קודמות של html.

ב- html5 לא משתמשים ב divים, אלא בתגיות סמנטיות.

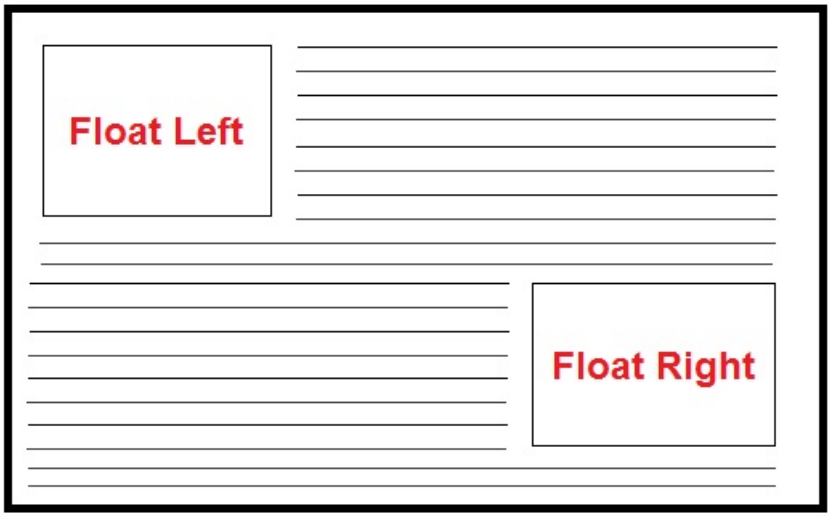
ציפת אלמנטים - **float** היא תכונה בשפת CSS המאפשרת לנו לשלוט במיקום אלמנטים מסוימים בדף. כאשר אנו מצמידים תכונה זו לאלמנט כלשהו התומך בה אנו בעצם מורים לדפדפן להכניס את האלמנט שלנו לשטף הרגיל של המסמך. כאשר הדפדפן מקבל הוראה להכניס אלמנט לשטף הכללי של המסמך הוא גורם לכל שאר האלמנטים בקרבתו של האלמנט להימנע מצד אחד מ"לדרוס" אותו ומצד שני מ"לדחוף" אותו. משמעות הדבר ששאר האלמנטים בדף יסתדרו סביב האלמנט בעל תכונת ה float.

איזה אלמנטים יכולים לצוף?

האלמנטים שיכולים לקבל את התכונה float הם אלמנטים מסוג קופסא. כלומר, אלמנטים שיוצרים סביבם בלוק. אלמנטים אלה יהיו כל התגיות הסמנטיות וכן תגית תמונה.

איך אלמנטים צפים?

התכונה float מאפשרת להגדיר ציפת אלמנטים רק לימין או לשמאל. תכונה זו אינה מאפשרת למשל ציפת אלמנטים למעלה או למטה. באיור הבא ניתן לראות דוגמה לציפת אלמנטים לשמאל ולימין:



דוגמה לציפה לימין
```
img {float:right;}
```
דוגמה לציפה לשמאל
```
section {float:left;}
```

התכונה float יכולה לקבל אחד מהערכים הבאים:

ערך	הסבר
inherit	האלמנט יורש את תכונות הציפה שלו מהאלמנט שבתוכו הוא מקנן.
left	האלמנטים משמאל לאלמנט המקבל ערך זה יצופו.
none	ערך ברירת המחדל. האלמנטים לפני ואחרי האלמנט המקבל ערך זה לא יצופו.
right	האלמנטים מימין לאלמנט המקבל ערך זה יצופו.

כיבוי ציפת אלמנטים

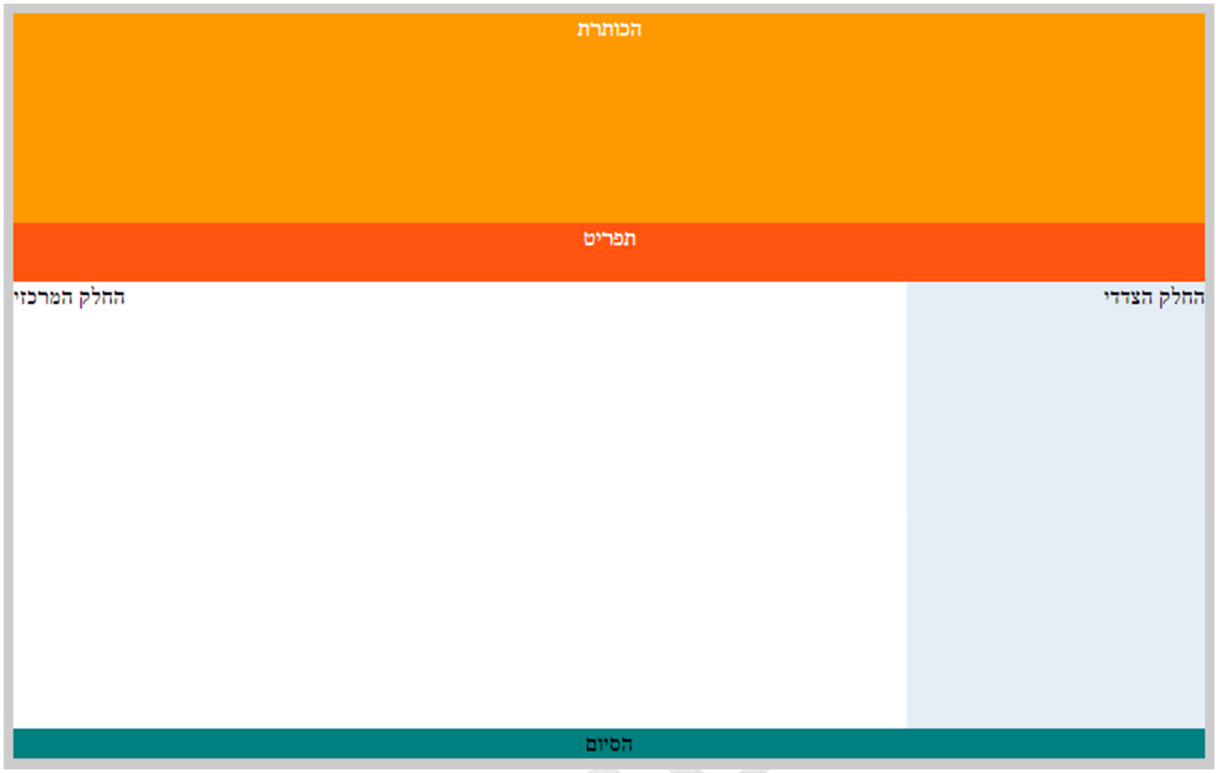
מרגע שאלמנט כלשהו קיבל את התכונה float אזי תכונה זו תגרום לציפה גם לאלמנטים הבאים אחריו. כדי למנוע את ציפת נשתמש בתכונה clear.

תכונה זו יכולה לקבל את הערכים הבאים:

ערכים	הסבר
both	לא תתאפשר ציפת אלמנטים מימין או משמאל לאלמנט המקבל ערך זה.
inherit	האלמנט יורש את תכונות הציפה או אי הציפה שלו מהאלמנט שבתוכו הוא מקנן.
left	לא תתאפשר ציפת אלמנטים משמאל לאלמנט המקבל ערך זה.
none	ערך ברירת המחדל. ציפת אלמנטים תתאפשר משני צדי האלמנט המקבל ערך זה.
right	לא תתאפשר ציפת אלמנטים מימין לאלמנט המקבל ערך זה.

דוגמא מלאה לסידור אלמנטים בדף:

התוצאה:



דף ה HTML:

```
<section id="wrapper">
      <header>הכותרת</header>
      <nav>תפריט</nav>
      <article>המרכזי החלק</article>
      <aside>הצדדי החלק</aside>
      <footer>הסיום</footer>
  </section>
```

קוד ה CSS:

```
body
 {
   background-color:#CCCCCC;
 }

 #wrapper
 {
   width: 800px;
   margin: 0 auto;
   background-color: white;
 }

 header
 {
   background-color:#FF9900;
   color: white;
   height: 140px;
   width: 100%;
```

```
  text-align:center;
}
nav
{
  background-color:#FF5511;
  color: white;
  height: 40px;
  width: 100%;
  text-align:center;
}
article
{

float:left;
width: 80%;
height: 300px;
background-color:White;
}
aside
{
  float:left;
  width: 20%;
  height: 300px;
  background-color :#E5EEF6;
  text-align: right;
}
footer
{
  width:100% ;
  background-color :Teal;
  text-align:center;
}
```

תגיות טקסט

תגיות אלו יתנו משמעות סמנטית לטקסט הנתחם בתוכם. התגיות יעזרו למנועי החיפוש להבין טוב יותר את תוכן האתר וכן יחזקו את פן השימושיות של האתר.

<time>

כשמה כן היא, תגית התוחמת בתוכה זמן או תאריך.

דוגמא:

```
~~<p class="pubdate">submitted on 02/23/10</p>~~
<p class="pubdate">  <time datetime="2010-02-10" pubdate>02/23/10</time></p>
```

<details>

תגית המכילה פרטים נוספים על אלמנט כלשהו. לתגית יש מאפיין open שמקבל true או false , ולפיו מוצגים או לא מוצגים הפרטים שתחומים בתגית זו. ניתן לצרף לתגית זו כותרת באמצעות התגית summary כותרת זו תשמש גם כפתור להצגה והסתרה של טקסט.

דוגמא:

```
<details open="open">
  <summary>The title of the details</summary>
  <p style="padding-left:20px">
    Content Content Content Content <br />
    Content Content Content Content <br />
    Content Content Content Content <br />
    Content Content Content Content <br />
  </p>
</details>
```

התוצאה:

▼ The title of the details

Content Content Content Content
Content Content Content Content
Content Content Content Content
Content Content Content Content

<mark>

בדומה לתגיות em ו- strong שתפקידן להדגיש, תגית זו תוחמת בתוכה טקסט מסומן.

דוגמא:

```
My Name Is: <mark>Shlomo</mark>
```

התוצאה:

My Name Is: Shlomo

<meter>

תגית המציינת מד התקדמות. כמו למשל במדריך התקנה או אשף הגדרות זה או אחר. התגית מכילה 2 תכונות: max שמכיל את מספר הצעדים המקסימלי ו-min שמכיל את מספר הצעדים המינימלי. למשל, אם יש לי מדריך מעמוד 1 עד עמוד 10 וכרגע אנו בעמוד 2, אני אכניס את הטקסט הבא כמציין מיקום:

דוגמא:

```
<meter max="10" min="0">2 from 10</meter>
```

התוצאה:

<progress>

תפקיד תגית זו הוא להציג התקדמות של תהליך. לרוב תתחום בתוכה אחוזים להתקדמות של הורדה או טעינה.

ההבדל בין meter ל progress:

התגית meter מכילה את התכונות **min** ו- **max**. ומציגה התקדמות בתכונה value, בין הערך המנימלי לערך המקסימלי שהוגדר.

לעומתה, התגית progress מכילה **רק** את התכונה **max**, ומציגה התקדמות בתכונה value בין 0 למקסימום שהוגדר.

תגית זו שימושית בעיקר להורדה והעלאה של קבצים באתר.

דוגמא:

```
<progress max="100" value="30"></progress>
```

התוצאה:

<output>

בדומה לתגיות code, ו-samp, תגית output תפקידה לייצג סוג של פלט, כמו כזה של קוד תכנות.

<dialog>

תגית שתוחמת בתוכה צ'אט. אם יש טקסט שמייצג שיחה בין אנשים רצוי לתחום אותו בתגית הזאת.

Datalist

תגיות טקסט ב-HTML 5

תגיות מולטימדיה

עד לתקן html5 שימוש במולטימדיה כדוגמת הפעלת וידאו, דרש שימוש בקוד סגור של טכנולוגיות שונות כדוגמת פלאש. קוד ה html היה ארוך ומסורבל, ולא היה חלק אינטגרלי מרשימת התגיות של השפה. התגיות החדשות בקטגוריה זו מאפשרות שימוש במולטימדיה באמצעות הדפדפן בעזרת קוד html פשוט.

<video>

תגית המציגה וידאו.

בדוגמא הבאה מגדירים אלמנט מסוג וידיאו בעל המאפיינים הבאים:

- **width**: רוחב הפקד
- **height**: גובה הפקד
- **autoplay**: הידיאו יופעל באופן אוטומטי, מיד עם עלית הדף.
- **controls**: המראה שלא אלמנט הוידיאו יוצג, כך שתהיה למשתמש שליטה על הוידיאו.
- **loop**: הוידיאו יתנגן שוב ושוב
- במידה והטעינה תכשל יופיע הטקסט המצורף.

התכונה source מגדירה את קובץ הוידיאו שיופעל:

- **src**: שם הקובץ שיופעל. movie.mp4
- **type**: סוג הפורמט

התגית **source** מופיע פעמיים. פעם ראשונה להגדרת הקובץ, ופעם שניה להגדרת פורמט אחר.

פורמטים נתמכים:
כיום נתמכים בעיקר הפורמטים הבאים:
MP4 container, H.264 video, all profiles, audio in AAC or MP3

שימי לב: יש לשלב קבצי וידאו או שמע, רק בפורמטים הנתמכים!

דוגמא:

```
<video width="320" height="240" controls="controls">
  <source src="movie.mp4" type="video/mp4" />
  <source src="movie.ogg" type="video/ogg" />
  Your browser does not support the video tag.
</video>
```

התוצאה:

<audio>

תגית שמע, פועלת כמו תגית וידאו ומשמשת לניגון קבצי סאונד.

מאפייני התגית זהים למאפייני התגית video.

תגית המציגה וידאו.

בדוגמא הבאה מגדירים אלמנט מסוג וידיאו בעל המאפיינים הבאים:

- **width**: רוחב הפקד
- **height**: גובה הפקד
- **autoplay**: קובץ השמע יתחיל להתנגן בצורה אוטומטית
- **controls**: המראה של אלמנט השמע יוצג, כך שתהיה למשתמש שליטה על ההשמעה.
- **loop**: הוידיאו יתנגן שוב ושוב

- במידה והטעינה תכשל יופיע הטקסט המצורף.

התכונה source מגדירה את קובץ הוידיאו שיופעל:
- **src**: שם הקובץ שיופעל horse.mp3
- **type**: סוג הפורמט

```
<audio id="audio1"  controls="controls">
<source src="horse.mp3" type="audio/mpeg" />
<source src="horse.ogg" type="audio/ogg" />
Your browser does not support the audio element.
</audio>
```

הפעלת / השהיית קובץ שמע באמצעות קוד js:
document.getElementById("audio1").play() / .pause()

<figure>

תגית זו משמשת על מנת לקבץ בתוכה תגיות מולטימדיה, כמו video ו audio.

ניתן להוסיף את תגית legend על מנת להציג כותרת לקבוצה זו.

<source>

תגית שמכילה URL לקובץ סאונד או וידאו. לרוב תופיע בתוך תגיות וידאו ואודיו.

<embed>

תגית מוכרת מהקוד של הצגת פלאש, אבל רק ב HTML 5-היא תיכנס באופן רשמי. תפקידה להציג תוכן מוטבע (חיצוני). לדוגמא:

<canvas>

קאנבס זוהי טכנולוגיה ליצירת גרפיקה בעזרת קוד, פרוט בהמשך.

INPUTE

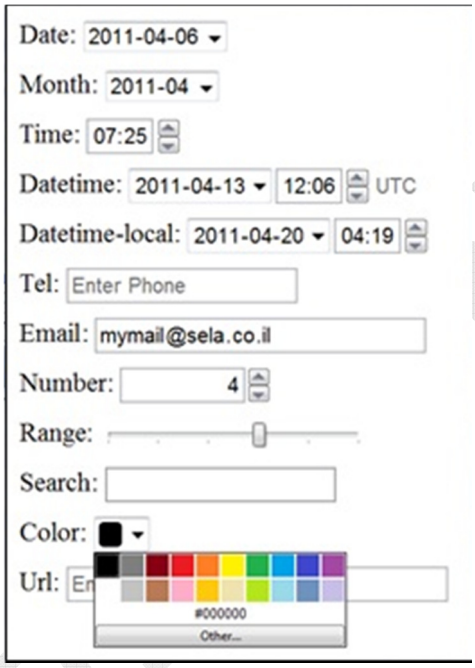
ב html5 נוספו סוגים חדשים של פקדי input, וכן תכונות נוספות לפקדים.

סוגים נוספים של פקדי Input

datetime-local,date,time,month,week,number,range,email,url,search,tel,color.

כדי להגדיר פקד שיהיה מטיפוס מסוים, יש לבחור את הטיפוס הרצוי בתכונה type:

```
<input type="color" value="red" />
```



מאפיינים חדשים לפקדי input

שם התכונה	הסבר	דוגמא
autocomplete	השלמה אוטומטית	
autofocus	הפוקוס יהיה בפקד מיד עם טעינת הטופס.	

סוגים שונים של input:

את סוג ה input הרצוי יש להגדיר בתכונה type.

formaction	מיועד עבור לחצן submit או image. מציין את השרת אליו ניתן לפנות.	
formmethod	מיועד עבור לחצן submit או Image. מציין את שיטת הגישה לשרת: get / post	
height and width	רוחב / גובה הפקד	`<input type="image" src="img_submit.gif" alt="Submit" width="48" height="48">`

List	התגית details המאפשרת השלמה אוטומטית של נתונים. התגית datalist אינה עומדת בפני עצמה, ויש לקשר אותה לפקד אחר, ע"י התכונה **list**. בדוגמא זו, תיבת הטקסט מקושרת ל datalist, ולכן בעת כתיבת תווים בתיבת הטקסט, תתאפשר בה השלמה אוטומטית של נתונים.	`<input list="browsers" name="browser">`
		`<datalist id="browsers">`
		`<option value="Internet Explorer">`
		`<option value="Firefox">`
		`<option value="Chrome">`
		`<option value="Opera">`
		`<option value="Safari">`
		`<details />`

min and max	מיועד לפקדי בעלי ערך מספרי כמו number או פקדי תאריך. מאפשר לקבוע ערך מינימלי ו/או ערך מקסימלי.	תאריך מקסימלי:
		`<input type="date" name="bday" max="1979-12-31">`
		תאריך מינימלי:
		`<input type="date" name="bday" min="2000-01-02">`
		טווח כמות:
		`<input type="number" name="quantity" min="1" max="5">`

Multiple	מיועד לפקדי email ו file. מגדיר האם יכול להכיל מספר מרובה של כתובות מייל או תאריך.	`<input type="file" name="img" multiple>`
pattern (regexp)	מגדיר ביטוי רגולרי לבדיקת תקינות של תוכן הפקד.	`<input type="text" name="country_code" pattern="[A-Za-z]{3}"`

title="Three letter country code">		
<input type="email" **placeholder**="הכנס כתובת מייל" /> הכנס כתובת מייל	מאפשר להגדיר טקסט כסימן מים המופיע לפני השימוש בפקד.	**placeholder**
form id="example" > action="HTMLPage2.htm" < <input type="email" **required="required"** placeholder="הכנס כתובת מייל" /> <input id="Submit1" type="submit" value="submit" /> <form/> הכנס כתובת מייל submit מלא שדה זה.	לחצן submit מעביר את הדף לשם הדף הכתוב בתכונה action של התג form. במקרה זה שם הדף הוא: HTMLPage2.htm. התכונה required מחייבת להכניס ערך לפקד. אם המשתמש לא הכניס ערך והוא מנסה לעבור לדף אחר, בדוגמא זו, ע"י לחיצה על submit, אך ניתן גם בדרכים אחרות, תופיע הודה שחובה למלא את השדה, והדף לא יעבור.	**Required**

תרגיל מס'1

צרי אתר ללימוד מקוון של שפת html.

דרישות התרגיל:

מבנה הדפים באתר:

בראש הדף לוגו של מגמת הלימוד, בצד ימין תפריט ניווט, בתחתית הדף פרטי יוצר האתר, במרכז הדף התוכן.

הדפים ישלבו שימוש בתגיות הסמנטיות הבאות: header, footer, nav, article

זכרי: התגיות הסמנטיות אינן משפיעות על העיצוב, יש להשתמש בעיצוב הדף ע"י התכונה float.

הדפים באתר:

- דף הבית – מאמר המתאר את הסטוריית ההתפתחות של שפת html (עייני בפרק המבוא), סרטון המציג נושא מתחום ה html (...)
- דף קליטת פרטי תלמיד – שם, כתובת, עיר, טלפון, כתובת מייל, תאריך התחלת הלימוד.
  הדף ישלב שימוש בתגיות הבאות: input, datalist.
  יש להגדיר אילו פרטים הם חובה באמצעות התגית mark.
  בתגיות input יש להשתמש בתכונה placeholder.
- דף שאלון אמריקאי - בדף יהיו 4 שאלות בתחום המחשבים, בכל שאלה ניתן לבחור תשובה 1 מתוך 3.

במקרה והתשובה נכונה השמיעי צליל עידוד. אין להציג את הממשק החיצוני של אלמנט השמע.

הדרכה:

באירוע onchange של התשובה הנכונה, הפעילי את אלמנט השמע ע"י הפונקציה play.

גרפיקה

Canvas

ראשי פרקים:

- אפשרויות הציור
- תכונות
- ציור בסיסי
- אפקטים

בגרסאות הקודמות של html לא ניתן היה ליצור ציורים בדף. אם היה צורך בציור של עיגול, היה ניתן להביא תמונה המכילה עיגול. הכתיבה באופן זה לא גמישה לחלוטין, כיון שלא ניתן לייצר את התמונות המתאימות בצורה דינמית, תוך כדי התוכנית.

ל- html5 יכולת ציור ע"ג הפקד Canvas. יכולת זה מהווה פריצת דרך משמעותית, והיא פותחת מרחב אפשרויות גדול בפיתוח אתרים.

ה - Canvas הוא בד ציור המאפשר לצייר עליו באמצעות קוד javascript.

אפשרויות הציור ב Canvas:

- ציור צורות בסיסיות כמו ריבוע, עיגול, קו וכדומה.
- הפעלת טרנספורמציות (שינויים) על הציור כמו סיבוב, שכפול אלמנטים, שקיפות, רקעים ועוד.

תכונות ה Canvas:

- הציור על ה - Canvas נשמר בזיכרון הנדיף. כלומר: לאחר שציירנו ריבוע, לא ניתן לשנות אותו. ניתן רק למחוק אותו (על ידי ציור ריבוע לבן מעליו) ולצייר את הריבוע הרצוי מחדש.
- ה - Canvas מצייר באמצעות פיקסלים (נקודות). שינוי ברזולוציה של המסך יעוות את הציור כמו כל תמונה המורכבת מפיקסלים.

ציור בסיסי:

שלב א:

בקוד ה - HTML נוסיף אלמנט Canvas. האלמנט נשאר **ריק**, והציור עליו מתבצע **רק** באמצעות פונקציות javascript.

```
<canvas id=" cnvs1" width="400" height="400"></canvas>
```

הערה: אין להגדיר את תכונות הגודל (width, heigth) בתוך style, אלא כתכונה של האלמנט.

שלב ב:

ציור על ה- canvas באמצעות פונקציות javascript. ניתן לצייר כל צורה בסיסית, ולהרכיב מהצורות הבסיסיות צורות חדשות.

הצורות שניתן לצייר: מלבן, עיגול, קו ישר, קו עקום ועוד.

להלן הסבר כיצד יוצרים פונקציה לציור על הקנבס, הפונקציה נכתבת היכן שנכתב כל קוד js.

ציור מלבן

דוגמא:

```
<script type="text/javascript">
        function drawRectangle() {
                var canvas = document.getElementById("cnvs1");
                var context = canvas.getContext("2d");
               context.beginPath();
                context.fillStyle = "red";
               var x = 20, y = 20, width = 150, height = 250;
                context.fillRect(x, y, width, height);
               context.closePath();
              }
</script>
```

הסבר הדוגמא:

קבלת האלמנט קנבס, עליו ניתן לצייר

var canvas = document.getElementById("cnvs1");

קבלת שטח הציור של הקנבס

var context = canvas.getContext("2d");

פונקציה המגדירה את התחלת הציור ומייצרת שטח לציור הנוכחי

context.beginPath();

הגדרת צבע המילוי

context.fillStyle = "red";

הפונקציה fillRect מציירת מלבן.

הפונקציה מקבלת את הפרמטרים הבאים:

1. מרחק נקודת ההתחלה מהגבול השמאלי של הקנבס
2. מרחק נקודת ההתחלה מהגבול העליון של הקנבס
3. גובה הצורה
4. רוחב הצורה

הגדרת הפרמטרים והצבת ערכים

var x = 0, y = 0, width = 150, height = 75;

קריאה לפונצקיה

context.fillRect(x, y, width, height);

פונקציה המגדירה את סיום הציור וסוגרת את השטח לציור הנוכחי

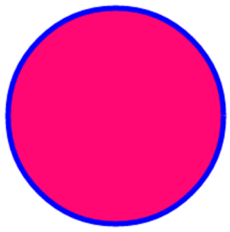
context.closePath();

התוצאה:

ציור עיגול

<u>דוגמא</u>:

```
function drawCircle () {
    var canvas = document.getElementById("cnvs1");
    var context = canvas.getContext("2d");
    context.beginPath();
    context.fillStyle = "#ff0873";
    var x=100, y=100, r=100;
    context.arc(x, y, r, 0, 2 * Math.PI);
    context.fill();
    context.lineWidth = 5;
    context.strokeStyle = "blue";
    context.stroke();
    context.closePath();

}
```



הסבר הדוגמא:

קבלת האלמנט קנבס, עליו ניתן לצייר

var canvas = document.getElementById("cnvs1");

קבלת שטח הציור של הקנבס

var context = canvas.getContext("2d");

פונקציה המגדירה את התחלת הציור

context.beginPath();

הפונקציה **arc** מציירת עיגול.

הפונקציה מקבלת את הפרמטרים הבאים:

1. מיקום נקודת המרכז בציר ה- x
2. מיקום נקודת המרכז בציר ה-y
3. אורך הרדיוס
4. נקודת ההתחלה של שרטוט העיגול

הערכים האפשריים:

0 – תחילת השרטוט בכיוון השעון, כאשר המחוגה עומדת על השעה 3 (אם משווים את המעגל לשעון).

מספר גדול מ 0 – תחילת השרטוט מנקודה אחרת במעגל הגבוהה מהשעה 3. הערך הגבוה ביותר הוא Math.PI*2 (קצת יותר מ 7).

מס' שלילי – כנ"ל אך בכיוון ההפוך.

5. נקודת הסיום של שרטוט העיגול

הערכים האפשריים:

Math.PI*2 – נקודת הסיום של השרטוט בכיוון השעון, כאשר המחוגה עומדת על השעה 3 (אם משווים את המעגל לשעון).

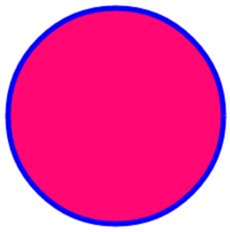
מספר קטן מ 7 – סיום השרטוט בנקודה אחרת במעגל הגבוהה מהשעה 3. הערך הגבוה ביותר הוא Math.PI*2 (קצת יותר מ 7).

מס' שלילי – כנ"ל אך בכיוון ההפוך.

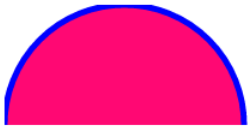
6. האם לשרטט כנגד כיוון השעון

דוגמאות לערכים בפרמטרים אלו:

עיגול מלא – context.arc(x, y, r, **0, 2 * Math.PI)**



חצי עיגול עליון – context.arc(x, y, r, **Math.PI, 2 * Math.PI)**



חצי עיגול תחתון – context.arc(x, y, r, **0, Math.PI)**

הגדרת הפרמטרים והצבת ערכים

var x = 0, y = 0, r = 100;

קריאה לפונצקיה

context.arc(x, y, r, 0, 2 * Math.PI)

הגדרת צבע המילוי

context.fillStyle = "#ff0873";

מילוי העיגול בצבע

context.fill();

ניתן להגדיר קו חיצוני לעיגול ע"פ השורות הבאות:

קביעת עובי הקו

context.lineWidth = 5;

קביעת צבע הקו

context.strokeStyle = "blue";

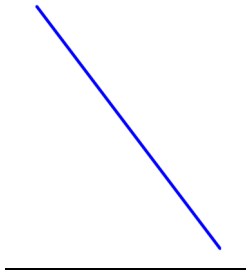
פונקציה יוצרת את הקו החיצוני ע"פ ההגדרות הנ"ל

context.stroke();

ציור קו

דוגמא:

```
function drawLine () {
                var canvas = document.getElementById("cnvs1");
                var context = canvas.getContext("2d");
               context.beginPath();
               var x=0, y=0;
               context.moveTo(x, y);
               var x =300, y = 350;
               context.lineTo(x, y);
               context.strokeStyle = "blue";
               context.lineWidth =20;
               context.lineCap = " square ";
               context.strock();
}
```



הסבר הדוגמא:

קבלת האלמנט קנבס, עליו ניתן לצייר

var canvas = document.getElementById("cnvs1");

קבלת שטח הציור של הקנבס

var context = canvas.getContext("2d");

פונקציה המגדירה את התחלת הציור

context.beginPath();

הפונקציה moveTo קובעת את נקודת ההתחלה של הקו ביחס לקנבס.

הפונקציה מקבלת את הפרמטרים הבאים:

1. מרחק נקודת ההתחלה מהגבול השמאלי של הקנבס
2. מרחק נקודת ההתחלה מהגבול העליון של הקנבס

הגדרת הפרמטרים והצבת ערכים

var x = 0, y = 0;

קריאה לפונצקיה

context.moveTo(x, y)

הפונקציה lineTo קובעת את נקודת הסיום של הקו ביחס לקנבס.

הפונקציה מקבלת את הפרמטרים הבאים:

1. מרחק נקודת הסיום מהגבול השמאלי של הקנבס
2. מרחק נקודת הסיום מהגבול העליון של הקנבס

הגדרת הפרמטרים והצבת ערכים

var x =300, y = 350;

קריאה לפונצקיה

context.lineTo(x, y)

קביעת עובי הקו

context.lineWidth =20;

קביעת צבע הקו

context.strokeStyle = "blue";

ניתן להגדיר את צורת הסיום של הקו ריבוע (square)/עיגול (round)

context.lineCap = " square ";

ציור הקו בפועל

context.strock();

ציור משולש:

```
context.beginPath();

context.moveTo(50, 100);
context.lineTo(100, 100);
```

```
context.lineTo(75, 25);

context.lineTo(50, 100);


context.fill();

context.stroke();

context.closePath();
```

ציור טקסט:

```
var c=document.getElementById("myCanvas");
var ctx=c.getContext("2d");
ctx.font="30px Arial";
ctx.fillText("Hello World",10,50);
```

התוצאה:

Hello World

אפקטים

מילוי מעבר צבעים:

הפונצקיות הבאות משמשות למילוי צורה במעבר צבעים:

- createLinearGradient(x,y,x1,y1) - מעבר צבעים קווי
- createRadialGradient(x,y,r,x1,y1,r1) – מעבר צבעים מעגלי

מעבר צבעים קווי:

```
var c=document.getElementById("myCanvas");
var ctx=c.getContext("2d");

var grd=ctx.createLinearGradient(0,0,150,80);
grd.addColorStop(0,"red");
grd.addColorStop(1,"white");
```

```
ctx.fillStyle=grd;
ctx.fillRect(10,10,150,80);
```

הפונצקיה createLinearGradient מגדירה אובייקט של מעבר צבעים

var grd=ctx.createLinearGradient(0,0,200,0);

פרמטרים:

הפרמטרים מגדירים את פריסת מעבר הצבעים ביחס לצורה.

2 פרמטרים ראשונים – נקודת ההתחלה של המעבר, בד"כ בתחילת הצורה: 0,0

2 פרמטרים אחרונים – נקודת הסיום של המעבר.

מעבר צבעים אלכסוני: ערכי הנקודות יהיו גודל הצורה (כמו בדוגמא)

מעבר צבעים אורכי: ערכי הנקודות יהיו רוחב הצורה ו- 0 (לפי הדוגמא: 150,0).

מעבר צבעים רוחבי: ערכי הנקודות יהיו 0 ורוחב הצורה (לפי הדוגמא: 0,80).

הפונקציה addColorStop מגדירה צבע מעבר ומקבלת את הפרמטרים הבאים:

1. מגדיר את נקודת העצירה של צבע המעבר. הערך יכול להיות 0-1
2. מגדיר את צבע המעבר

grd.addColorStop(0,"red");
grd.addColorStop(1,"white");

הפקודה הבאה מגדירה את צבע המילוי של הריבוע, באובייקט מעבר הצבעים, ומכאן המשך הציור כרגיל.

ctx.fillStyle=grd;

מילוי צורה במעבר צבעים מעגלי:

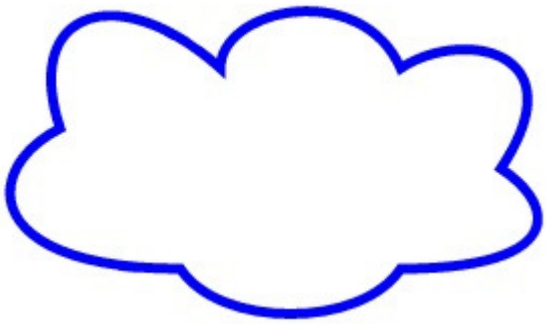
```
var c=document.getElementById("myCanvas");
var ctx=c.getContext("2d");

var grd=ctx.createRadialGradient(75,50,5,90,60,100);
grd.addColorStop(0,"red");
grd.addColorStop(1,"white");
```

```
ctx.fillStyle=grd;
ctx.fillRect(10,10,150,80);
```

דוגמאות נוספות:

הקוד הבא יצייר מין צורה של ענן.

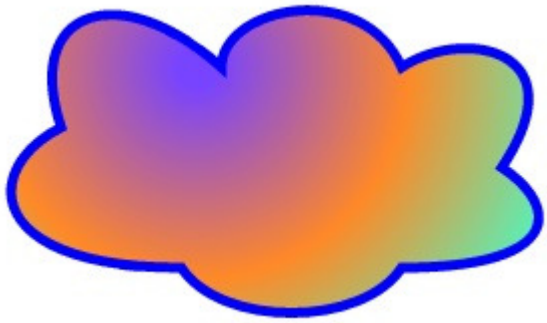
```
// begin custom shape
ctx.beginPath();
ctx.moveTo(170, 80);
ctx.bezierCurveTo(130, 100, 130, 150, 230, 150);
ctx.bezierCurveTo(250, 180, 320, 180, 340, 150);
ctx.bezierCurveTo(420, 150, 420, 120, 390, 100);
ctx.bezierCurveTo(430, 40, 370, 30, 340, 50);
ctx.bezierCurveTo(320, 5, 250, 20, 250, 50);
ctx.bezierCurveTo(200, 5, 150, 20, 170, 80);
ctx.closePath();
// complete custom shape
ctx.lineWidth = 5;
ctx.strokeStyle = "#0000ff";
ctx.stroke();
```



כעת אם נוסיף (לפני השורה של lineWidth) את השורות הבאות:

```
var grd = ctx.createRadialGradient(238, 50, 10, 238, 50, 200);

grd.addColorStop(0, "#7944FF"); // light blue

grd.addColorStop(0.5, "#FF8928"); // light blue

grd.addColorStop(1, "#42FFCC"); // dark blue

ctx.fillStyle = grd;

ctx.fill();
```

נקבל:



סיכום:

Canvas הוא אחד היכולות החזקות של HTML5, הוא מאפשר לייצר גרפיקות בצורה שניתן היה לבצע בעבר רק באמצעות flash או silverlight.

תרגיל מס' 2

צרי מסמך HTML שתפקידו ללמד תווי נגינה בסיסיים.

א. הציגי 7 לחצנים עבור כל תו.
ב. בלחיצה על לחצן, יושמע התו המתאים ויופיע איור בסיסי של התו במיקום הנכון בחמשה.

הנחיות:

1. פונ' תקבל כפרמטר את פקד הצליל, תפסיק את נגינת הצליל הקודם ותנגן את הצליל המבוקש.
2. פונ' לניקוי ה canvas וציור החמשה מחדש.
3. פונ' לציור תו: קו ועיגול בכל פעם במקום המתאים, לפי פרמטרים שתקבל. (טיפול מיוחד בתו דו).

SVG – גרפיקה וקטורית

SVG - Scalable Vector Graphics זוהי טכנולוגיה ליצירת גרפיקה מבוססת XML (שפת תגיות) שקל להציג בה גם אנימציות, והיא אינה תלוית רזולוציה.

בדומה ל-HTML 5 Canvas, גם עם SVG ניתן להציג צורות וגרפיקה. אך בניגוד ל-Canvas המשתמש בפונקציות JavaScript, ב SVG עובדים עם תגיות.

ההחלטה שלנו אם לבחור ב-SVG או Canvas נובעת בדרך כלל מאופי השימוש. אם אנו צריכים פתרון דינמי – קוד שנוצר תוך כדי ריצה (לדוגמא, בהתאם לבחירת המשתמש) נבחר ב-Canvas. פתרון SVG יהיה טוב אם אנו צריכים הצגת גרפיקה סטטית יחסית שמתאים לכל רזולוציה.

SVG מאפשרת עריכת אנימציות בקלות יחסית, והיא כוללת מגוון רחב של אפשרויות גרפיות.

ההבדלים בין Canvas ל SVG:

SVG	Canvas
לאחר יצירת הצורה, היא מהווה חלק מאלמנטי DOM, ניתן לגשת אליו בקוד js וניתן לקשר אותו לאירועים.	הציור על canvas מורכב מפיקסלים ואינו מהווה אובייקט. לא ניתן לפנות אליו בקוד js או לקשר אותו לאירועים.
כאשר משתנה אחת התכונות של האלמנט, הדפדפן ישנה את הצורה באופן אוטומטי.	הצורה על Canvas נוצרת ע"י פיקסלים והיא אינה זכורה לדפדפן. כל שינוי מצריך ציור מחדש של הצורה.
איטי יותר לרנדרור	ניתן לשמור את הציור על ה canvas כקובץ תמונה.
לא מתאים למשחקים	מהיר יותר לעיבוד ומתאים למשחקים עתירי גרפיקה.

בפרק זה אנו נכיר רק את הצורות הבסיסיות ואפשרויות המילוי של SVG.

התגית svg:

תגית זו מגדירה את משטח הציור של הצורה.

כל צורה חייבת להיות בתוך תגית זו, ניתן להגדיר מספר צורות בתגית svg בודדת.

```
<svg width="320" height="320">
```

הערה: התגית svg אינה יכולה להפעיל אירועים. כדי לקשר את הצורה לאירוע, יש למקם את התגית בתגית נוספת, כג' section.

תגיות צורה:

רוב הצורות מכילות אחת או יותר מהתכונות הבאות:

- fill – צבע רקע
- stroke – צבע קו המסגרת

- width – רוחב
- height – גובה
- x, y – נקודת ההתחלה של הצורה ביחס למשטח הציור svg.
- cx, cy – נקודת המרכז לעיגול / אליפסה.
- stroke-width – עובי קו המסגרת

ישנן תכונות נוספות, המתאימות לצורה הספציפית, כג' רדיוס לעיגול, נקודות למצולע ועוד.

מלבן:

```
<svg width="320" height="320">
        <rect fill="orange"
          stroke="black"
          width="150"
          height="75"
          x="50"
          y="25" />
</svg>
```

עיגול:

r - רדיוס

```
<circle fill="orange"
          stroke="black"
          stroke-width="3"
          cx="40"
          cy="25"
          r="20"/>
```

אליפסה:

rx – רדיוס לרוחב

ry – רדיוס לגובה

```
<ellipse fill="orange"
          stroke="black"
          cx="100"
          cy="60"
          rx="75"
          ry="50"/>
```

קו ישר:

```
<line x1="30"
          y1="30"
          x2="150"
          y2="85"
          stroke="red"
          stroke-width="4"/>
```

מצולע פתוח:

points – תכונה זו מכילה סדרת נקודות. כל נקודה מורכבת מ x, y. קווי המצולע עוברים מנקודה לנקודה.

בדוגמא זו נוצר משולש.

```
<polyline points=" 0 0 ,50 0, 50 50"
          fill="orange"
          stroke="black"
          stroke-width="4"/>
```

מצולע סגור:

points – תכונה זו מכילה סדרת נקודות. כל נקודה מורכבת מ x, y. קווי המצולע עוברים מנקודה לנקודה.

בדוגמא זו נוצר משולש.

```
<polygon points=" 0 0 ,50 0, 50 50"
          fill="orange"
          stroke="black"
          stroke-width="4"/>
```

טקסט:

```
<text x="0" y="15" fill="red">I love SVG</text>
```

מילוי צבעים:

מעבר צבעים קווי:

defs – הגדרת מעבר הצבעים.

stop – מגדיר צבע מעבר ואחוז השטח עד אליו הוא מתפרס.

ניתן לקבוע את כיוון המעבר, ע"י שימוש בתכונות X1, Y1, X2, Y2.

נקודות אלו מציינות את תחילת המעבר, וסוף המעבר.

לדוגמא: למעבר אלכסוני נגדיר את הערכים

x1=0 y1=0 x2=1 y2=1

לאחר הגדרת מעבר הצבעים, נגדיר צורה והיא תקבל את אובייקט המעבר.

```
<svg width="320" height="320">
        <defs>
            <linearGradient id="MyGradient">
                <stop offset="33%" stop-color="yellow" />
                <stop offset="66%" stop-color="orange" />
                <stop offset="99%" stop-color="red" />
              </linearGradient>
        </defs>

        <rect fill="url(#MyGradient)" stroke="black" stroke-width="5"
        x="20" y="20" width="300" height="100"/>
```

```
</svg>
```

מעבר צבעים מעגלי:

```
<svg width="320" height="320">
        <defs>
            <radialGradient id="RadialGradient1">
                <stop offset="10%" stop-color="yellow" />
                <stop offset="90%" stop-color="blue" />
            </radialGradient >
        </defs>

        <rect fill="url(#RadialGradient1)" stroke="black" stroke-width="5"
        x="20" y="20" width="300" height="100"/>
</svg>
```

מילוי תבנית:

```
<svg width="320" height="320">
        <defs>
            <radialGradient id="RadialGradient2">
                <stop offset="10%" stop-color="yellow" />
                <stop offset="90%" stop-color="blue" />
            </radialGradient>
            <pattern id="mycircleandsquare" patternUnits="userSpaceOnUse"
            x="0" y="0" width="150" height="100">
                <rect fill="url(#MyGradient)" stroke="black" stroke-width="5"
                 x="20" y="20" width="30" height="30"/>
            </pattern>
       </defs>
        <ellipse fill="url(#mycircleandsquare)" stroke="black"
        stroke-width="5" cx="400" cy="200" rx="350" ry="150" />
    </svg>
```

אפקטים:

צל:

```
<svg >
        <defs>
            <filter id="f1" x="0" y="0" width="200%" height="200%">
                <feOffset result="offOut" in="SourceGraphic" dx="20" dy="20" />
                <feBlend in="SourceGraphic" in2="offOut" mode="normal" />
            </filter>
        </defs>
        <rect width="90" height="90" stroke="green" stroke-width="3"
        fill="yellow" filter="url(#f1)" />
    </svg>
```

טשטוש:

```
<svg>
        <defs>
            <filter id="f1" x="0" y="0">
                <feGaussianBlur in="SourceGraphic" stdDeviation="15" />
            </filter>
        </defs>
        <rect width="90" height="90" stroke="green" stroke-width="3"
        fill="yellow" filter="url(#f1)" />
    </svg>
```

תרגיל מס' 3

צרי את אחד הציורים הבאים בעזרת Svg:

א. שמים זרועים כוכבים עם ירח לא מלא.

ב. מראה של שקיעה: רקע בגווני מעבר מתאימים, ושמש.