תוכן עניינים

[**HTML** 2](#_Toc454971631)

[מידע כללי ותכונות 2](#_Toc454971632)

[תגיות ומאפייני אלמנטים 3](#_Toc454971633)

[טבלאות - Table 4](#_Toc454971634)

[FORM 5](#_Toc454971635)

[Frame / iFrame 6](#_Toc454971636)

[Applet 6](#_Toc454971637)

[Meta tags 6](#_Toc454971638)

[שאלות למבחן בנושא HTML 6](#_Toc454971639)

[**CSS - Cascading Style Sheets** 7](#_Toc454971640)

[תכונות CSS 7](#_Toc454971641)

[**עקרונות JavaScript** 9](#_Toc454971642)

[תכונות JAVASCRIPT 9](#_Toc454971643)

[DOM - Document Object Model: 13](#_Toc454971644)

[**JQuary** 15](#_Toc454971645)

[**HTML 5** 16](#_Toc454971646)

[**AJAX** 19](#_Toc454971647)

[תכונות AJAX 20](#_Toc454971648)

[Object Oriented on Java Script 21](#_Toc454971649)

[**JSON – JavaScript Object Notation** 22](#_Toc454971650)

[תכונות JSON 22](#_Toc454971651)

[JSP - Java Server Page 23](#_Toc454971652)

[Servlets: 23](#_Toc454971653)

[JSP Syntax 24](#_Toc454971654)

[**Cookies and Session Management** 24](#_Toc454971655)

[Session Management 25](#_Toc454971656)

[Server Side 26](#_Toc454971657)

[**JavaBeans** 26](#_Toc454971658)

[שאלות 28](#_Toc454971659)

# HTML

## מידע כללי ותכונות

**HTML** ((**HyperText Markup Language** שפת תגיות (לא שפת תכנות!) לתצוגה ועיצוב דפי אינטרנט ותוכן לתצוגה בדפדפן. זו שפת המרכזית בעולם האינטרנט, המהווה שלד למרבית עמודי התוכן באינטרנט. השפה מאפשרת עיצוב תוכן בצורה מהירה, קלה ללימוד באופן יחסי וקלה לכתיבה. מטרתה היא להגיד לדפדפן איך להציג נתונים!

HTML תוכננה לעבוד על כל מחשב, מכל סוג והיא סלחנית מאד לגבי פרטים קטנים. היא מותרת לשימוש על ידי כל מפתח אתרים, ללא צורך ברכישת זכויות יוצרים מחברה כלשהי, והיא ניתנת לקריאה בכל סוגי המערכות. HTML מאפשרת ליצור קישורים למסמכי HTML נוספים או לסוגים שונים של נתונים. כמו כן, היא תומכת במולטימדיה, היא מאפשרת להציג נתונים בעזרת: טקסט, צליל, תמונה, וידאו ועוד.

**!** שפת HTML היא טקסט בלבד, אך באמצעותה ניתן להציג מולטימדיה.

**!** attributes - מאפיינים של התגית נמצאים אך ורק בתגית הפותחת.

**! אלמנט element-**

------------------------------------------

<html>

<head>

<title> Site Title </title>

</head>

<body>

</body>

</html>

------------------------------------------

תכונות HTML:

* HTML הוא הסטנדרט בדפי האינטרנט
* בנוי על TAGS עם אלמנטים בתוכם
* לא case sensitive
* ישנם 3 חלקים לאלמנט : תג תכונות, התוכן, סיום התג.
* <html> ... </ Html> אלמנט השורש של המסמך כולו. ממוקם על הקווים הראשון והאחרון של המסמך.
* <head> ... </ Head> מספק מידע תיאורי על מסמך כחלק בכותרת.
* <title> ... </ Title> מוצג בחלק העליון של המסך. ממוקם בתוך התג head. <body>
* ... </ Body> מזהה את התוכן של דף האינטרנט. רוב התגים נמצאים בתוך תגית זו.
* 2 פורמטים של תמונות נתמכים בHTML : מוכר על ידי סיומת הקובץ. Gif. תומך 8 סיביות צבעים. Lossless דחיסה - איכות התמונה ללא שינוי. מתאים תמונות אמנות קו.

מוכר על ידי סיומת הקובץ. JPEG. 8 ביט או 24 ביט צבע תואם. קטן יותר מאשר קובץ GIF של אותה תמונה. מתאים תמונות אמנות

* מה קובע את התצוגה של הטקסט? HTML מיועד להציג את הטקסט זוהי השפה. כל המאפיינים האחרים מוסיפים לייצוג. המטרה היא לקבוע איך להציג נתונים.
* השפה היא סבלנית. היחידה בעולם המחשוב שהיא סבלנית היא ייחודית.
* HTML זה רק טקסט בכתיבה שלו אבל להצגה הוא יכול להציג תמונה והכל.

## תגיות ומאפייני אלמנטים

|  |  |  |
| --- | --- | --- |
| פתיחה וסגירה של מסמך html | <html>…</html> | |
| בין פקודות אלו יוגדר כל מה שאינו גוף המסמך כגון: כותרות וקשרים למסמכים אחרים | <head>…</head> | |
| כותרת המסמך שתופיע בשורת הכותרת של הדפדפן | <title> … </title> | |
| בין פקודות אלו יוגדר גוף המסמך. | <body>…</body>  bgcolor="" //Background color | |
| הוספת שורה חדשה | <br> | |
| הוספת קו הפרדה | <hr> | |
| רשימות ממוספרות, ניתן לקבוע מספר התחלה ע"י קביעת start="num", | 1. Coffee 2. Tea    * Milk | <ol start="1">  <li>Coffee</li>  <li>Tea</li>  <ul><li>Milk</li></ul>  </ol> |
| קישור לאתר | <a href="http://SiteAddress.com">Link Text</a> | |
| בניית כותרת, כאשר 1 היא הגדולה ביותר ו 6 קטנה ביותר. יורד שורה אוטומטית בהוספת הכותרת. | <h?>. . . </h?> // ? = 1-6 | |
| אובייקט תמונה (gif/jpg), יכול להציג גם וידאו, ניתן לערוך את גודלה של התמונה, במידה ולא יוגדר התמונה תוצג בגודלה המלא. | <img src="smiley.gif" alt="Smiley face" height="42" width="42"> | |
| מאפייני טקסט | <b>text</b> -**Bold Text**  <p>text</p> - normal text (Paragraph)  <i>text</i> - *inclined text*  <u>text</u> - under Line | |
| טבלאות:  table – יוצר טבלה, ניתן לקבוע את מאפייני הטבלה  tr – TableRow שורה חדשה  td – TableData תא בשורה | <table border="1" style="width:100%">   <tr>     <td>Jill</td>     <td>Smith</td>      <td>50</td>   </tr>   <tr>     <td>Eve</td>     <td>Jackson</td>      <td>94</td>   </tr> </table> | |
| iframe – מטרתו להציג דף html בתוך מקום שמוקצה בדף html נוכחי | code:  <iframe width="100%" height="300px" src="demo\_iframe.htm" name="iframe\_a"></iframe>  attribute for Link to change iframe page:  target="iframe\_a" | |

ישנם 5 תווים שלא ניתן לכתוב אותם כי הם הדפדפן אינו יודע א להציג או שהם שייכים לקוד. לדוגמא: >. לכן נשתמש ב:

&nbsp; -- non breaking space

&lt; -- less then

&gt; -- grather then

&amp; -- ampersand

&quot; -- quotation

**מאפייני אלמנטים:**

* בונה שורה חדשה ללא תנאי <br>
* קו הפרדה זה <HR>
* האם האלמנטים הם סמנטיים? לא! (האם אומרים שיש את המשמעות בתוכם הכוונה לשמות משמעותיים ). דוגמא לאלמנט סימנטי : <date> סימן שזה תאריך אבל בHTML יש רק אלמנטים בלי שמות משמעותיים כמו br,p,ul וכו'...
* לכל נתון יש שני היבטים: 1. משמעות שלו 2. איך להציג אותו.
* IMG האם הוא יכול להציג תמונה? כן אבל הוא לא כולל את התמונה.
* <IMG> יכול להציג גם וידאו!
* אלמנט קישור <a>
  + אפשר לעשות לינקים בתוך האתר?כן.
  + אפשר לעשות לינקים למסמך אחר? כן.
* שאלה שיהיה בה אלמנט ואז אלמנט ההפעלה יכול להיות מה נכון ומה לא נכון?
* אלמנט של HTML מתחיל עם **התחלה תג / תג הפתיחה**
* אלמנט של HTML מסתיים עם תג סגירה.
* **אלמנט תוכן** זה הכול בין ההתחלה ועד הסוף של התג
* אלמנטים מסוימים של HTML מכילים **תוכן ריק**
* אלמנטים ריקים **נסגרים בתוך תג פתיחה**
* רוב מרכיבי ה-HTML יכול להכיל **תכונות**
* **רוב אלמנטים של** HTML **יכולים להיות מקוננים** (**יכולים להכיל אלמנטים** HTML **אחרים**).
* מסמכי HTML מורכבים מאלמנטים של HTML מקוננים.
* האלמנטים **h6, h5, h4, h3, h2, h1** משמשים ליצירת כותרות (h זה קיצור של "heading" - כותרת), כאשר h1 היא הדרגה הראשונה ובאופן טבעי הטקסט הגדול יותר, h2 היא הדרגה השנייה ולכן טקסט קטן במעט ו-h6 זו הדרגה השישית והאחרונה בהירארכיית הכותרות ולכן גם הטקסט הקטן ביותר.
* ul> > זה בעצם מחלק את זה לרשימת סעיפים
  + **<li> - זה בעצם בולט.**
  + **- <ol> סעיפים מספריים אבל לעשות את זה מחוץ ל<li>**

## טבלאות - Table

<table>

<th> מסמל כותרת של כל עמודה בטבלה.

<tr> </tr> זה יורד שורה-מסמל שורה

tr = table row

<td> מסמל תא

td = table data

* **מה התפקיד של הטבלה?** האם אפשר להשתמש בטבלאות ואפשר לקבוע מיקום יחסי של אובייקטים? כן!
* איך אפשר לקבוע מיקום של כל טבלה בדפדפן? לפי הסדר של הכתיבה כל עוד שיהיה מקום. **סוף תג / תג הסגירה**
* אפשר לקבוע באמצעות טבלה את מיקום האובייקט מימין או משמאל וכו'...
* בנוי משורות ועמודות.
* עיצוב טבלאות יעשה ע”י שימוש ב- CSS בלבד, אולם פעמים רבות בשלב בניית הדף, נרצה לראות את הטבלה בבירור. לשם כך נשתמש במאפיין border.
* התא הוא הצטלבות של שורה ועמודה.
* תאים מכונים על ידי שורה, עמודה
* בתוך כל תא מידע, רושמים את המידע שלנו.
* אסור לכתוב דברים בין ה TR ל TD !
* הטבלה לא נותנת לנו גבולות בברירת מחדל. גבולות הם Border. צריך לרשום <table border=”1”> למשל כדי לראות גבול.
* אם לא קובעים גודל אז הטבלה מתאימה עצמה לגודל המסך.
* אז אם אני רוצה שבטוח אובייקט אחד יהיה ליד השני, משתמשים בטבלה.
* Rowspan – ממתח השורה , Columnspan - ממתח העמודה
* מאפייני הטבלה: width, height, border, align, bgcolor, bordercolor, cellspacing (המרווח בתוך התאים \ בין האובייקטים), cellpadding (המרחק של הטקסט מגבול התא).
* טבלאות מגדרות על ידי **<table>** תג. טבלה מחולקת לשורות (על ידי **<tr>** תג), וכל שורה מחולקת לתאים (על ידי **<td>** תג). אותיות **td** ראשי תיבות "table data," שהוא התוכן של תאים.
* תאים של טבלה יכולים להכיל טקסט, תמונות, רשימות, פסקאות, קווים אופקים, טבלאות, וכו
* אם אתם לא מציינים תכונות גבול טבלה תוצג ללא שום גבולות. לפעמים זה יכול להיות שימושי, אבל רוב הזמן, אתם תרצו שגבולות יוצגו
* תאי טבלה ללא תוכן לא יוצגו כהלכה ברוב דפדפנים.. '

**שאלה למבחן בנושא טבלה:** מה קורה אם כמות התאים שונה בין השורה הראשונה לשנייה? תהיה מסגרת אבל בתא יהיה אפור.

## FORM

* בתוך הטופס אני מבקש לשים קלטים.
* את הטופס השרת מקבל. התוכנה תעבד נתונים ותחליט מה לעשות הלאה

<form action=” “>

Name:

<input type=”text”>

<input type=”submit“>

</form>

* זה שולח לאן שאני אומר לו את המידע.
* לפורם אין ייצוג על המסך

**שאלה**- כמה כפתורי submit יכולים להיות בפורם אחד? תשובה: כמה שבא לי

**שאלה** – כמה טפסים יכולים להיות במסמך אחד? תשובה: כמה שבא לי

* אם עושים אינפוט בלי פורם, אז אי אפשר לשלוח את זה לשום מקום.

**<input>**

* מאפשר ליוזר להכניס טקסט. יש מאפיינים.
* Hidden – מאפשר למשתמש לשלוח נתונים בסתר.
* <button> - כפתור
* לינקים: מוגדרים ע"י הפקודה A שהיא קיצור ל Anchor.
* ניתן לעשות לינק לתוך האתר שלי, אבל חייב להשתמש ב Relative.
* לא משתמשים ב MailTo.

## Frame / iFrame

* מסגרות. זהו מנגנון להצגת מס' דפי HTML על חלון אחד.
* Frameset – מגדיר את החלוקה של הדף. מחליף את ה Body (עמ' 47). <כוכבית זה מה שנשאר>
* Inner frame - IFRAME – תופס שטח שאותו מגדירים, ומציג בו את הדף שמבקשים. זה נוח כי לעתים יש תוכן שבו רוצים להשתמש בכמה מקומות. זה טוב כי אני יכול לשנות את התוכן מתי שאני רוצה בחלק ספציפי של הדף.

## Applet

<Applet)..</Applet> מאפשר הכנסת קוד ג'אווה לתוך קובץ HTML. כבר לא נמצא בשימוש זאת בעקבות שימוש בפלאש. כמו כן גם הפלאש יצא משימוש בעקבות HTML5.



## Meta tags

מרכיב HEAD מכיל מידע כללי על מסמך

תפקיד ה META היא לספק META INFO על העמוד

## שאלות למבחן בנושא HTML

1.שאלה למבחן: מסמך HTML הוא מסמך טקסט בלבד? כן

2.האם תצוגת מסמך HTML זה טקסט בלבד? לא

3. האם במקום הזה נהוג להשתמש Absolute Path? תשובה: לא, כי כל הסיפור צריך להיות נייד.

.4 האם ניתן לעשות לינק בתוך אותו דף? תשובה: כן

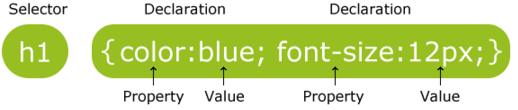
איך?

<a href = “#p1”> to first Paragraph </a>

כמובן שלפני זה צריך לסמן <p id = “p1”>

**אם יש סולמית למקום לא מסומן, הוא פשוט הולך להתחלת המסמך. זה בגלל שהשפה טולרנטית לשגיאות**

# CSS - **C**ascading **S**tyle **S**heets



מיועד לקבוע עיצוב של הדף.

**החלת קובץ סטייל חיצוני:** כדי לקבוע שהדף יהיה בעיצוב ע"פ קובץ ה css עלינו להוסיף:

<head>  
<link rel="stylesheet" type="text/css" href="mystyle.css">  
</head>

**הוספת סטייל לקובץ:** כמו כן, ניתן להוסיף קוד Style ב CSS בדף עצמו ע"י הוספת <style> … </style> בתוך ה HEAD ובין תגיות אלה להגדיר את העיצוב לדף. עיצוב זה לא יחל גם על דפים נוספים ולכן לא נהוג להשתמש בזה.

**יצירת סטייל חדש:** ניתן לצור סטייל חדש ע"י כתיבת .Name , ואז באלמנט לכתוב class="Name". דוגמא:

<style> p.outset {border-style: outset;} </style>

<p class="outset">An outset border.</p>

**הוספת סטייל בתוך אלמנט:**

<h1 style="color:blue;margin-left:30px;">This is a heading.</h1>

* הסטייל שיקבע הוא הסטייל הספציפי ביותר שיצרנו.
* ניתן להוסיף לקובץ מספר קבצי סטייל.
* הסטייל מאפשר למפתחים לשלוט בעיצוב של מספר דפים. כיום קובעים את עיצוב הדף רק באמצעות CSS.
* סלקטורים:
  + element החלת ה CSS על כל סוגי האלמנטים
  + #id – החלת ה CSS על אלמנט לפי ID
  + .class / element.class – החלת ה CSS על כל האובייקטים מסוג CLASS מסוים.

## תכונות CSS

* מגדירים סגנונות לדפי HTML .
* היא לא שפת תכנות.
* Selector{name:value};
* הסלקטור מיועד להגיד למי מיועד הסגנון.
* סלקטורים יכולים להיות שלושה סוגים:
* #id שם המזהה אחרי הסולמית.
* div. לכתוב את שם האלמנט ואז נקודה ואז את שם המזהה של המחלקה.
* div{name:value} אפשר לכתוב את שם האלמנט ואז הוא פונה לכולם מסוג האלמנט הזה.
* יש הגבלות על המזהה בכל השפות אי אפשר לכתוב בשם של מזהה סימן שאלה ורשתית אלא רק באותיות ומספרים וכמות מוגבלת של סימנים מיוחדים כמו קו תחתון ומקף.
* כדי למקם את האלמנטים באופן שונה ממה שמסודר בדרך כלל אלמנט אחרי אלמנט נשתמש בתכונה של **;position: absolute**, ובנוסף נגדיר היכן יימצא האלמנט ביחס לדפדפן. לשם כך אפשר להשתמש באחת האפשרויות הבאות:top,left,…
* css מוגדר כסטיילים מדורגים, רשימת עיצובים מדורגים. כי אפשר להגדיר אותם במקומות שונים.
* 1. HEAD,BODY
* 2. קובץ עם סיומת CSS ולקשר אותו לדף הנוכחי.
* 3. בתוך כל אלמנט אפשר לכתוב למשל <p style={color=red…
* כמה מקומות אפשר לכתוב ?CSS רק בשלושה מקומות.
* אם רשמתי בשלוש מקומות לאותו נתיב אז יהיה את הכי קרוב... בתוך האלמנט עצמו ואז יהיה בכותרת ואז בקובץ החיצוני.
* כברירת מחדל הצבע הוא שחור.
* האם כל הדפדפנים מראים אותו דבר ב?css כן אבל לא תמיד! **(אם כתוב תמיד זה לא נכון).**
* CSS מגדיר כיצד להציג רכיבי HTML מאפשרת למפתחים לשלוט בסגנון ובפריסה של דפי אינטרנט מרובים בבת אחת.

**הגדרת רווחים Padding**

ה- padding הוא הרווח בין התוכן עצמו (זה יכול להיות טקסט, תמונה או כל אובייקט אחר) לבין הגבול (border) שלו. חשוב להגדיר padding על מנת שהתוכן (למשל טקסט בתוך תא בטבלה) לא יהיה צמוד לקו הגבול. צבע הרווח יהיה כצבע הרקע של האלמנט. עובי ה- padding מוגדר בפיקסלים, בסנטימטרים או באחוז מתוך גודל האובייקט.

**הגדרת שוליים Margins**

בניגוד ל- padding שהוא רווח סביב האובייקט בתוך הגבולות, margin הוא רווח מחוץ לגבולות. הרווח משמש על מנת ליצור הפרדה בין אלמנט אחד למשנהו בעמוד אינטרנט.

ברוב המקרים, אין צורך להגדיר שוליים מכל צדי האובייקט.

* **BORDER**

על מנת להגדיר כיצד ייראה הגבול (border) סביב אלמנט כלשהו בעמוד אינטרנט, קיימות

ב-CSS תכונות לעיצוב עובי הגבול, הצבע ועוד, אך לפני הכול, יש להגדיר את סוג הגבול באמצעות התכונה bordr-style. ללא הגדרת תכונה זו, אף אחת מן התכונות האחרות שנגדיר עבור הגבולות לא תבוא לידי ביטוי.

באמצעות CSS ניתן לעצב את כל אחד מצידי הגבול בנפרד.

* תחביר CSS מורכב משלושה חלקים: בורר(סלקטור), מאפיין וערך: בורר {נכס: ערך} בורר הוא בדרך כלל רכיב HTML שאתה רוצה להגדיר, המאפיין הוא התכונה שברצונך לשנות ולאחר כל מאפיין יכול לקחת ערך.
* עם סלקטור מחלקה ניתן להגדיר סגנונות שונים עבור אותו סוג של רכיבי HTML.
* אתה יכול גם להשמיט את שם התג בבורר כדי להגדיר סגנון שישמש כל רכיבי HTML בעלי מחלקה מסוימת.
* סלקטור id שונה מסלקטור מחלקה. בעוד סלקטור מחלקה עשויים לחול על כמה אלמנטים בדף, סלקטור מזהה תמיד חל על רכיב אחד בלבד. תכונת זיהוי חייב להיות ייחודי במסמך.
* <head><style></style></head> Head style:
* קובץ חיצוני : <head>
* <link rel="stylesheet" type="text/css" href="definition.css"> </head>
* כל הסגנונות יעלו אל דף חדש "וירטואלי" לפי כללים ברורים, מן העדיפות הנמוכה ביותר אל העדיפות הגבוהה ביותר: דפדפן ברירת המחדל. גיליון עיצוב חיצוני. סגנון דף פנימי (בתוך התג head). סגנון בתוך השורה (בתוך אלמנט ה-HTML)

# עקרונות JavaScript

כדי להפעיל סקריפט עלינו להכניס אותו לתוך תגיות <script></script>.

השפה אומנם מזכירה JAVA אך שונה. לצורך העניין, אין צורך להצהיר על משתנים. משתמשים בשפה זו לבדיקות של קלטים, שינוי עיצובים והפעלת סקריפטים פשוטים.

מטרת השימוש:

1. הגבה לבקשות של צד לקוח.

2. לבדוק תקינות של טפסים.

3. שינוי דף באופן בינאמי.

4. שליטה באפלטים של ג'אווה ופלאגאינים של הדפדפן.

השפה היא Case Sensitive, ותומכת ב OOP.

כל הסקריפטים בדף הם אותו סקריפט, כך שהמשתנים שהוגדרו בפונקציה אחרת בדף מוכרים על-ידי פונקציות אחרות.

השפה נקראת loosely-typed, אינו מצריך להכריז על סוגי המשתמשים אך קביעת סוג המשתנה נקבעת בשמה.

ישנם חמישה סוגים של משתנים: מספרים (אין הפרדה אין int ל float), משתנה בולאני, מחרוזות (אין characters – תו בודד הוא מחרוזת בגודל 1), סוג נתונים null, סוג נתונים undefined (ערך שעדיין לא אותחל לסוג משתנה מסוים).

**Flow Control:**  ניתן לכתוב פקודות if,else,while,for,break,continue.

## תכונות JAVASCRIPT

* גאווה סקריפט היא שפת תכנות
* ג'אווה סקריפט מונחה עצמים!!!
* ניתן לכתוב הערות (אותו סינטקס כמו ג'אווה).
* כן case sensitive
* היא לא תת קבוצה של ג'אווה
* לא חייבים לרשום (;) נקודה פסיק בסוף המשפט בגאווה כי זה עדיין יהיה תקין גם בלי..
* אין קומפיילר אין מי שעושה את ההמרה.
* אפשר לשנות עיצובים באמצעות גאווה סקריפט.
* השפה הזאת היא object oriented. Oop
* יש קונסטרוקטור בגאווה.
* יש new בגאווה.
* אין CLASS בגאווה.
* מגדירים אובייקטים חדשים בעזרת קונסטרוקטור.
* יש מילה שמורה THIS משתמשים בקונסטרוקטור.
* איפה אפשר לכתוב גאווה סקריפט? באלמנט סקריפט.
* אפשר כמה פעמים שרוצים סקריפטים.
* יש סדר מוגדר וחד משמעי בגאווה.
* מופעל לפי סדר הכתיבה,
* <script> ו </ SCRIPT> בכל מקום בדף.
* רוב המאפיינים הם לקריאה בלבד.
* src - ניתן לשינוי מאפשר להחליף תמונה באופן דינמי ע"י הצגת האובייקט שלה. תכונה חשובה מאוד! ניתן להשתמש אנימציות. החלפת תמונה על פי קלט משתמש. וכו ' עובד הכי טוב אם התמונה החדשה היא באותו גודל כמו המקורי. עוד תמונה חדשה תותאם לגודל המקורי.
* אובייקט מבוסס, מונחה עצמים ניתן ליצור ולהשתמש באובייקטים, אבל אין תמיכה מיוחדת פולימורפיזם וכו '
* - DOM Document Objects Model אובייקטים של DOM המתארים מסמך מורכב ומגדיר גישה לאותם אובייקטים JS.

אובייקטים חשובים: החלון. המסמך. מיקום. ההיסטוריה

* JavaScript היא תחבירית במשפחת C השוני העיקרי מC הוא במערכת שבה ניתן להתייחס לפונקציה כערך!
* יש ירושה – לא קלאסית רגילה עם מתודות וקלאסים אלא שונה – עושה את הכל אותו דבר ואפילו נותנת אפשרויות שליטה גדולה יותר – היא מתייחסת להכל כאובייקטים, ולכן קל יותר.
* כל האובייקטים קשורים ישירות או בעקיפין ל- Object.prototype כל האובייקטים יורשים כמה שיטות בסיסיות. אף אחת לא ממש שימושית, אין להעתיק שיטות. אין את שיטת EQUALS.

**יצירת אובייקט ב JAVASCIPT**

* ליצור אובייקט חדש - לשלושת הביטויים האלה יש בדיוק את אותה תוצאה:

**Using NEW:** var person = new Object();

**Using Object Literal:** var person = {firstName:"John", lastName:"Doe", age:50}

**Using Object Constructor:**

function person(first, last, age, eye) {  
    this.firstName = first;  
    this.lastName = last;  
    this.eyeColor = eye;}

var myFather = new person("John", "Doe", "blue");

**HTML AND JAVA SCRIPT**

* הגאווה מלווה את הhtml .
* עושים פעולות בhtml בעזרת גאווה אפשר לעשות מניפולציות.
* לשנות נתונים, לשנות תצוגה, לעשות אנימציה וכו'... אפשר לעשות עם גאווה...

בגאווה אפשר לעשות הכל רק שיהיה בhtml

* לגאווה יש אינטראקציה גדולה עם HTML.
* ברגע שפתחתי סקריפט זה כבר גאווה ולא HTML ושם חלים רק החוקים של גאווה.
* שליטה בסביבה הדפדפן והתכנים בו . מאפשר מניפולציה של האובייקטים בדף ה-HTML. מאפשר שינוי דור דינמי / של HTML התסריט בדרך כלל מוטבע ישירות לתוך דף ה-HTML.
* מתן אינטראקציה פשוטות בדפי אינטרנט. מגיב לפעולות הדפדפן של המשתמש, כגון לחיצה על קישורים, עוזב דף, וכו ' אימות קלט הטופס בצד הלקוח. שינוי דפים באופן דינאמי מחושב על סמך נתונים. על פי גרסת הדפדפן. בהתבסס על תצורת הדפדפן. וכו ' שליטה יישומוני Java ותוספות בדפדפן.
* בגלל הרצף שבו HTML עובד להגדיר פונקציות ומשתנים גלובליים בסעיף <HEAD>.
* Image נוצר בדרך כלל עם תג <img> ב-HTML. תמונות ניתן לתת תכונת id. לאחר מכן ניתן להתייחס לתמונה לפי ערך id

<body>

<IMG id= "pic1" src="images/tiger.gif">

<script>

alert(pic1.src);

// pic1, refer to the image

</script>

</body>

**משתנים**

* האם למשתנים של JS יש סוג? כן!
* האם רושמים מראש את המשתנים? לא!
* האם אפשר לשנות באמצע התוכנה את המשתנים ? כן!
* אחת השאלות: האם המשתנה יכול לקבל ערך של פונקציה (אפילו פונקציה שלא נתתי לה שם)? כן!
* Var x; לא מוגדר אבל מגדיר תוך כדי.
* המשתנים הם אותם משתנים. אבל לפני הסקריפט הם לא מוכרים.
* משתנים בתוך הפונקציה- אם יש לפני המשתנה VAR זאת אומרת שהמשתנה הזה הוא מקומי רק בתוך הפונקציה ולא מחוץ לפונקציה. אם לא רשמנו VAR אז אפשר להשתמש במשתנה בכולם.

**פונקציות**

* מילת מפתח FUNCTION.
* יכולים להיות פרמטרים.

אפשר להפעיל את הפונקציה עם כמה פרמטרים. אפילו 4 פרמטרים.

* + אם ניקח 4 פרמטרים הוא ייקח את השניים הראשונים.
  + אפשר גם בלי פרמטר. וברירת המחדל זה לא מזוהה לא מוגדר

**Undefined**

* על מנת לדעת אם התו (@ לצורך העניין) נמצא בטקסט אז משתמשים בפונקציה הזאת. בהתחלה כותבים את שם המזהה שרוצים לחפש בו.

id\_name.value.indexOf("@")

* איפה אפשר לכתוב פונקציה? באלמנט סקריפט
* הפונקציות מוכרות בכל הדף.
* פונקציות עשוי לקחת מספר של ארגומנטים טיעונים לא מאופיינים, עובר תמיד על ידי ערך. לקרוא לפונקציה באמצעות האופרטור () למשל, () functionName להציג רשימה של כל ארגומנטים בתוך הסוגריים.
* פונקציית למספר **Number(**value**)** ממירה ערך למספר. הוא מייצר NaN אם יש לו בעיה. דומה ל אופרטור prefix +.
* לא חייב להגדיר פונקציות מראש, אפשר להגדיר פונקציות בתוך פונקציות כך שברגע שפונקציה א' תפעל, היא תפעיל מתוכה את פונקציה ב' גם אם לא הוגדרה לפני כן בתוכנית
* מכיוון שפונקציות הם ערכים, פונקציות ניתן לאחסן באובייקטים. פונקציה בתוך האובייקט נקראת שיטה.
* אם קוראים לפונקציה עם יותר מדי ארגומנטים, היא תתעלם מהארגומנטים הנוספים אם קוראים לפונקציה עם מעט מדי ארגומנטים, אז הערכים החסרים יהיו **undefined**.
* ישנן ארבע דרכים לקרוא לפונקציה:
* functionObject(arguments( בצורת פונקציה
* thisObject.methodName(arguments) בצורת שיטה
* thisObject["methodName"](arguments)
* new functionObject(arguments) בצורת בנאי
* *functionObject***.apply(***thisObject***,**  בצורת החלה

**[***arguments***]**

* פונקציה יכולה לקבל מספר אין סופי של משתנים
* המשתנים שמתקבלים אינם צריכים להיות מוגדרים ע"י סוג המשתנה אלה רק השם שלו
* פונקציה לא צריכה להצהיר על סוג הערך שהיא מחזירה או אם היא מחזירה ערך כלשהו
* פונקציות יכולות להיות מועברות כפרמטר בתוך פונקציה
* פונקציות הם גם אובייקטים
* פונקציה יכולה להיות מוגדרת בתוך פונקציה קיימת

**מערכים**

* משתמשים בסוגריים מרובעים [] לגישה של אלמנטים למערך
* להתחלת המערך משתמשים ב() new array
* למערך יש תכונה בשם length
* מערכים הם typeless. כל רכיב יכול להיות סוג אחר.
* אפשר לקבל מערכים של מערכים.
* אינדקסים מתחילים מ 0.
* מערכים הם דלילים ייתכן שיהיה "חורים".
* ערכים לא מוגדרים יש הערך המיוחד null.
* המאפיין אורך הוא המדד הגבוה ביותר בתוספת 1​​.
* גודל ניתן לשנות באופן דינמי.

**var arr1 = new Array();**

**Var anArr=[];**

* יוצר מערך חדש
* Join([XXX,YYY,ZZZ]) --- באמצעות מפריד המצורף או פסיק, מצרף את כל המרכיבים של המערך למחרוזת אחת.
* מה חשוב לגבי המערך בגאווה? שהם לא רצופים בזיכרון.
* מערכים מוגדרים כ - loosly-typed המשמעות של זה היא שכל ערך במערך יכול להיות מסוג שונה.
* פעם שנותנים ערך לאלמנט במערך, אי אפשר למחוק אותו, אפשר לשנות את הערך שלו ל NULL
* כל פעם שאנחנו נותנים ערך לתא מסוים גודל המערך גדל באחד לדוגמא:



* + כאשר אנחנו מגדירים את a[i+1] . המערך אוטומטי מגדיל את הגודל של ל 11
* כאשר מוחקים ערך במערך הוא נהיה undefined.

**אובייקטים**

* אם יש לי אובייקט ויש לו משתנה בשם איקס אז אפשר לכתוב a.x=8;
* אז האם אפשר לכתוב ככה? A["x"]=8כן גם אפשר! סוגריים מרובעים!
* גישה לנתונים חברי מחלקה אפשר לקבל גם באמצעות אופרטור [ ]
* אם אין A אובייקט כזה הוא עושה שגיאה.
* אם לא היה X משתנה כזה בתוך האובייקט אז הוא יוצר אותו.
* אובייקטים אפשר להעביר כארגומנטים לפונקציות,
* והוא יכול להחזיר על ידי פונקציות אובייקטים,
* אובייקטים לא מועברים לפי ערך.
* אופרטור === משווה הפניות אובייקט, לא ערכים , מחזיר נכון רק אם שני הגורמים להשוואה הם אותו אובייקט.

# DOM - Document Object Model:

קובע איך מסמך HTML הופך לאוסף אובייקטים.

DOMהנו ממשק לתכנות יישומים (API), המאפשר ליישומים ותסריטים (script) לגשת ולעדכן באופן דינמי את התכולה, המבנה והסגנון של מסמכי HTML ו־XML.

לדוגמה, אחד העצמים המרכזיים הקיימים הוא עצם ה־document, המייצג את המסמך כולו, ואשר מסוגל להכיל בתוכו עצמים נוספים בהתאם לתכולת המסמך, כמו תיבת טקסט, כפתור או שדה למילוי בטופס.

שימוש באובייקטים מקל באופן משמעותי על מפתחי אתרים המתכנתים ב-JavaScript.

ה -DOM נתמך כיום על ידי כלל הדפדפנים העיקריים בשוק.

ממשק ה־ DOM מייצג את המסמך תוך שימוש בתכנות מונחית העצמים, כך שכל אלמנט במסמך מיוצג כעצם (אובייקט) בעל תכונות (properties) ושיטות/פעולות (methods).

* document.getElementByTagName('tag') פנייה לאלמנטים בHTML בגישה ישירה ללא צורך בהגדרת מזהה

שאלה: האם כל דפדפן עושה שימוש אחיד בכל סגנון? **לא!,** לכן משתמשים ב DOM כדי שהמסמך יופיע באופן אחיש בכל הדפדפנים.

בראשי תיבות **DOM**, הנו ממשק לתכנות יישומים (API), המאפשר ליישומים ותסריטים (script) לגשת ולעדכן באופן דינמי את התכולה, המבנה והסגנון של מסמכי [HTML](http://he.wikibooks.org/wiki/HTML) ו־[XML](http://he.wikibooks.org/wiki/XML).

לדוגמה, אחד העצמים המרכזיים הקיימים הוא עצם ה־document, המייצג את המסמך כולו, ואשר מסוגל להכיל בתוכו עצמים נוספים בהתאם לתכולת המסמך, כמו תיבת טקסט, כפתור או שדה למילוי בטופס.

 שימוש באובייקטים מקל באופן משמעותי על מפתחי אתרים המתכנתים ב-[JavaScript](http://he.wikibooks.org/wiki/JavaScript).

 ה -DOM נתמך כיום על ידי כלל הדפדפנים העיקריים בשוק.

 ממשק ה־DOM מייצג את המסמך תוך שימוש בתכנות מונחית העצמים, כך שכל אלמנט במסמך מיוצג כעצם (אובייקט) בעל תכונות (properties) ושיטות/פעולות (methods).

* document.getElementByTagName('tag') פנייה לאלמנטים בHTML בגישה ישירה ללא צורך בהגדרת מזהה.

ה DOM הוא בעצם אוסף אובייקטים של הדפדפן בעם פונקציות:

alert, blur, close, focus וכן הלאה...

**DHTML:**

שילוב של CSS, HTML ו JS שנותנת לשנות כל דבר בתוך הדף ע"י JS. היכולת לשנות כל סטייל שמוגדר ב CSS. זאת ע"י שינוי של הסטייל באמצעות פונקציה. הבעייתיות בשימוש ב DHTML היא שלא כל הדפדפנים מייחסים באופן זהה למאפיינים של ה DOM ולפקודות JS ומוסיפים מאפיינים ופקודות שאינן קיימות בתקן על מנת ליצור אפשרויות חדשות. דבר זה מקשה כעל בניית דף המתאים לכל הדפדפנים.

כמו כן, נוספו 2 אלמנטים:

**<div>**

* אלמנט מסוג גוש.
* תוכן מהרבה סוגים, כולל טקסט.
* מאפיינים כלליים (ליבה, שפה ואירוע).
* תגית זו למעשה מגדירה אלמנט שאיננו אלמנט "סטנדרטי" (כגון פיסקה, כותרת, טבלה, וכו').

למעשה, בקוד שלנו, רוב התגיות יהיו תגיות div ולא תגיות סטנדרטיות, כאשר אלו יציינו אובייקטים כמו: תפריט, לוגו, טופס צור קשר, וכו'

**<span>**

* מאפשר לעבוד עם מחרוזות תוך התייחסות לאות בודדת.
* אלמנט מסוג שורה.
* תוכן מסוג טקסט או שורה.
* מאפיינים כלליים (ליבה, שפה ואירוע(
* תגית זו מגדירה חלק מאלמנט על מנת לעצבו.

# JQuary

* jQuery: לכתוב פחות, לעשות יותר,
* jQuery בספריית JavaScript היא סוג חדש בספריית JavaScript. המפשטת במהירות יצירת מסמך HTML, טיפול באירוע, Animating, ואינטראקציות AJAX לפיתוח אינטרנט מהיר.
* צורת כתיבה $("blablabla")
* ספריית פונקציה שהיא מקלה בפיתוח גאווה סקריפט.
* היא לא הספרייה היחידה שיש בשוק. אחד מכמה.
* הדולר זה השם של הפונקציה וגם השם של האובייקט.
* דולר מקבל סלקטור ושם אפשר לכתוב $(“a:first”) הוא פונה לאיבר הראשון שנקרא a. נין לבחור מס' אובייקטים באמצעות שרשור של פסיק.
* **כל מה שכותבים ב JQUERY ניתן לכתוב גם ללא שימוש ב JQUERY, פשוט הקוד יהיה יותר ארוך!**

**האם דולר מחזיר מערך או אובייקט שעוטף מערך אלמנטים? הוא מחזיר אובייקט עוטף מערך אלמנטים של HTML.**

* $(“a:first”) . באמצעות נקודה אחרי אפשר להשתמש בפונקציות של מובנות בג'יי קוורי

**API –** ספריית פונקציות. בכל שפה ישנם API רבים המובנים לעבודות שונות כגון עבודה עם קבצים, שליפת נתונים וכן הלאה... JQUERY הוא API לעבודה עם JS בסביבת HTML.

**יתרונותיה העיקריים של jQuery:**

* גודל קוד – קובץ ה- include קטן (24KB מוקטן ומכווץ ו- 77KB מוקטן ללא כיווץ).
* יעילות – הפעולות הן מהירות ויעילות בביצועיהן.
* תיעוד ותוספים - עקב הפופולאריות הרבה ש- jQuery צברה בשנים האחרונות, התיעוד שקיים עבורה רחב ביותר ומספר התוספים שנכתבו עבורה הוא רב.
* בחירת אלמנט DOM באמצעות דפדפן מנוע קוד פתוח Sizzle, פרויקט שולי של jQuery
* שינויים של DOM, כולל תמיכה עבור CSS 1-3 ו-XPath בסיסי
* JQUERY תומך בכל הדפדפנים הפונקציות שכתובות בו תוכמכות בכל חמשת הדפדפנים הפופולריים ביותר (כרום, אקספלורר, אופרה, פיירפוקס, ספארי..).
* מניפולציה CSS – ניתן לערוך את הסטייל של אובייקט מסויים ע"י שימוש ב JQ.
* JavaScript תוספים
* $("p").click(); פונקציה רגילה
* $("p").click(function(){ פונקציה מורחבת  
    // כאן נכתבת הפונקציה רגיל כמו JavaScript  
  });

**יתרונות jQuery**

* גודל הקוד קטן
* מהירות ריצה טובה מאוד
* מאוד פופולרית היום
* מספר תוספים עשיר
* תיעוד מפורט ובהיר
* "write less, do more"

**מה jQuery מאפשר לעשות**

* בחירה של אלמנט או קבוצת אלמנטים מתוך דף html
* שינוי אלמנט/קבוצת אלמנטים ע"י שינוי הגדרות ה-css
* הכנסת אלמנט חדש לדף (לדוגמה חלון של הודעה או טופס)
* מחיקת אלמנט מדף
* הוספת פונקציונליות ajax לדף - ניתן לבצע קריאה בפרוטוקול get או post ולקבל את תוצאת פונקצית ה-ajax כטקסט רגיל, או כאוביקט בפרוטוקול json
* הגדרת פונקציות כתגובה לארועים שונים הפועלים על אלמנט מסוים (או קבוצה). ב-jquery מוגדרים מראש כ-20 ארועים בהם ניתן להשתמש (לדוגמה: dblclick - הקלקה כפולה על אלמנט, change - בחירת אופציה מתוך תפריט select, ועוד.
* אנימציה ואפקטים

כמו כן, קיימות הרחבות רבות שמספקות רכיבים שלמים מן המוכן כמו תפריטים מורכבים, גלריית תמונות, active grid - מאפשר להציג מידע טבלאי ולערוך אותו במקום ועוד...

**אירועים:**

הגדרת פונקציות כתגובה לארועים שונים הפועלים על אלמנט מסוים (או קבוצה). ב-jquery מוגדרים מראש כ-20 ארועים בהם ניתן להשתמש (לדוגמה: dblclick - הקלקה כפולה על אלמנט, change - בחירת אופציה מתוך תפריט select, ועוד.

**מאפיינים נוספים:**

* 1. כאשר אנו מכניסים למשתנה פונקציה, הפונקציה תפעל רק אחרי שכל הדף ייטען

*$(function(){*

*code here will execute after DOM is ready*

*});*

כמו כן ניתן לכתוב על פונקציה .ready ובכך הפונקציה תפעל אחרי שהדף ייטען במלואו.

* 1. ניתן לבצע אנימציות באמצעות JQ ע"י שינוי מיקום אובייקט.
  2. JQUERY כמו JS נטען ראשון בדף, לכן בפונקציות שמחילות מאפיינים שונים על אובייקטים יש לכתוב אותם אחרי האובייקטים או להשתמש במאפיינים 1 ו 2 לעיל.

# HTML 5



XHTML – שפה רגישה לשגיאות שפה זו התפתחה מתוך רצון לבצע ניתוח מידע מאתרים אחרים. לדוגמא כדי לבנות תוכנה שאוספת נתוני מניות מאתרים אחרים. דוגמא נוספת היא מנוע חיפוש, מנוע חיפוש אוסף מידע מאתרים אחרים.

Semantic Web: לפי רעיון זה הומצא ה XHTML שהיא טולרנטית לשגיאות.

HTML – גרסה חדשה של תקן HTML שיצאה בשנת 2014 במטרה לשפר וליצור תקן אחיד ל HTML. מאפשרת לפתח WEB APPLICATION דרך הדפדפן.

**דברים שנוספו ל HTML5:**

* ציור ב HTML
* אנימציות
* שפורים ל CSS כגון שימוש ב GRADIENT
* ניתן לבנות משחקים
* אפשר לקבל גישה לרכיבים במכשיר כגון מיקרופון, GPS, מצלמה וכן הלאה.
* ניתן לעבוד עם האתר/אפליקציה גם במצב מקוון (מנותק). כל זאת ע"י שמירת נתונים על המכשיר עצמו.
* יצירת קשר ושליחת נתונים (Web Socket) ללא שימוש בשרת.
* נוספו אלמנטים למולטימדיה (וידאו ואודיו).
* שיפורים נוספים לגבי ביצועים וריצת התוכנה.

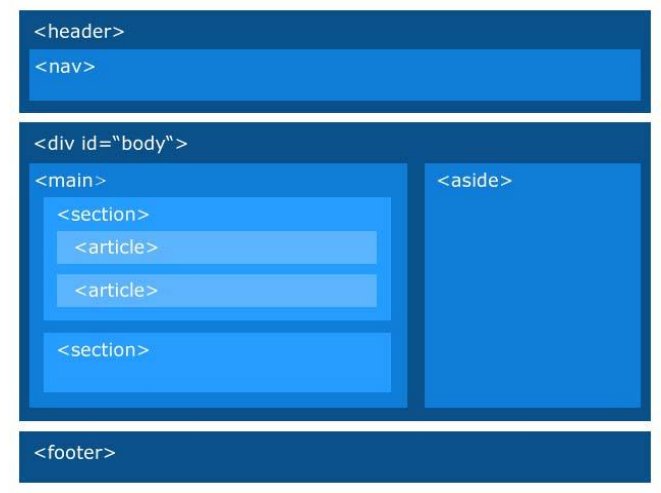
**הצהרה על HTML 5** – לא כמו HTML בגרסאות קודמות, כאן אין צורך לכתוב את הגרסה ואפשר והדפדפן יודע לזהות שהשימוש הוא HTML5.

<!DOCTYPE html>

**קידוד:**

פשוט יותר: <meta charset="utf=8">

**סגירת תגיות:** המון תגיות אינן צריכות תגית סגירה כגון TR,DD,TH,TD, וכן הלאה...



**אלמנטים חדשים:**

אלמנטים שמטרתם לעצב ולסדר ולחלק את הדף בצורה לוגית נכונה.

header – שם האתר ושם החברה

nav – מקום לשים את התפריט והקישורים לדפים אחרים

section – מקטע של מידע. לדוגמא באתר של חדשות, הנושא ספורט יכול להיות ב section נפרד.

article – תוכן שהוא בלתי תלוי בתוכן אחר שנמצא באותו דף. בשונה מ section שהמידע שמוצג בו הוא תלוי במידע אחר.

**חשוב להבין –** article ו section הם אובייקטים בעלי אותה רמה היכולים להכיל זה את זה.

footer – החלק התחתון בדף, לרוב מכיל קישורים, מידע כללי וזכויות יוצרים של הדף. בדרך כלל חוזר על עצמו מדף לדף

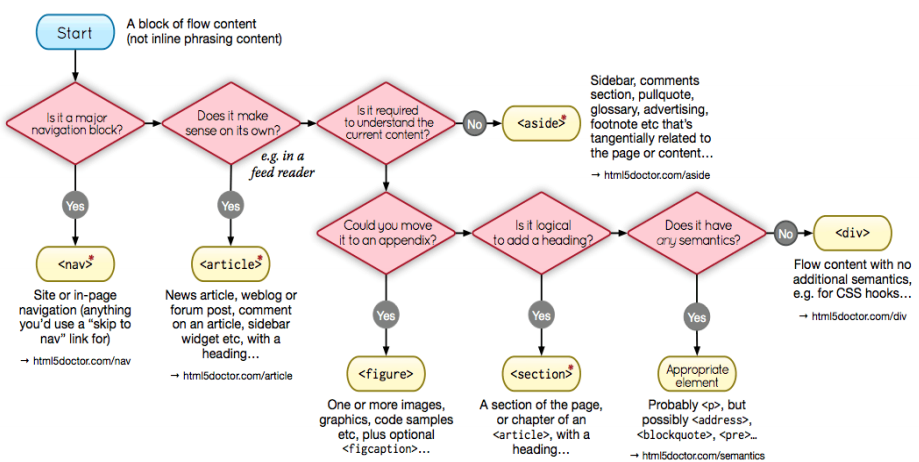
time – אלמנט חדש המתאר זמן ותאריך. יכול לשמש אפליקציות למציאת התאריך בדף בקלות וכמו כן לשנות את התצוגה לפי זמנים שונים כמו בוויז שמשתנה בלילה.

aside – תוכן שלרוב לא קשור לתוכן הראשי כגון פרסומות, קישורים וכן הלאה... לרוב נמצא בצד.

figure – אובייקט שמשמש לבנות אובייקט אחד מכמה אובייקטים כגון: תמונה משולבת (קולאז), טבלה + כותרת וכן הלאה...

main – יכול להיות רק main אחד בדף והוא התוכן המרכזי של הדף. לא יכול להיות בן של article, aside, footer, header, nav.

h1-h6 כתיבת הכותרות ב HTML5 משמשת לייצוא תוכן עניינים (מבצע את זה בכותרות h1-h3.



**אלמנטים מ HTML4 שעברו שינוי / חידוש סמנטי:**

b - מילות מפתח ושמות.

i – מושגים טכניים

hr – פסקה חדשה שבה מתבצעת החלפת נושא

br – עדיך להשתמש רק כשחייב, עדיפות לבצע את המרווח ב CSS אבל אם צריך שורה חדשה כמו בכתיבת כתובת אז ניתן להשתמש

title – ניתן להוסיף לאלמנט, משמש כדי במערכת תוכל להקריא בקול את המידע על האלמנט. דוגמא, בעל מוגבלות יכול ללחוץ על התמונה ושהמערכת תקריא לו את התוכן של התמונה.

details/summary – מאפשר כתיבת כותרת בתוך וכאשר לוחצים עליה נפתח טקסט שכתוב בתוך.

**שיעור 2:**

השיומש ב HTML עוזר לדפדפן להתאים את אתר האינטרנט לשימוש במובייל ובמכשירים אחרים, זאת ע"י חלוקה לוגית נכונה של הדף.

אתר לבעלי מוגבלויות:

label – עוזר לנגישות בתוך האתר (בטפסים) לבעלי מוגבלויות, ניתן להגדיר קריאה של הכיתוב בקול, ומעביר את הפוקוס ל INPUT בלחיצה על שם השדה.

הוספת אינפוטים חדשים ל HTML5:

צבע, מייל, מספר, טווח, חיפוש, טלפון, כתובת, תאריך, תאריך וזמן, חודש, שבוע, זמן, תארי וזמן מקומי.

האינפוטים ב HTML5 חל עליהם מנגנון בדיקה אוטומטי, הבודק את תקינות הנתונים שהוכנסו ע"פ תבנית. לדוגמא בודק בכתובת המייל שמכילה דומיין ושטרודל.

את הבדיקות ניתן לבצע ע:י תבנית שאותה אנו כותבים לדוגמא:

קוד מדינה

pattern="[a-z]{3}"

באינפוט זה ניתן להכניס אותיות קטנות באנגלית ועד 3 אותיות. במידה ומוסיפים \* ניתן לכתוב עד 3, במידה ורושמים + חייב יותר מתו אחד. בדוגמא רשום מספר מדויק.

# AJAX

Asynchronous Javascript And XML

למעשה יכולת לקבל ושלוח לשרת נתונים ללא ביצוע מעבר לדף הבא. AJAX הוא למעשה תופסת ל JAVASCRIPT והיא מספקת אינטגרציה ברמה (LEVEL) גבוהה עם המשתמש. בשונה מ DCHTML , AJAX תקשורת עם השרת ללא מעבר לדף הבא.

דוגמא, בגוגל, כאשר אנו מבצעים חיפוש אנו רושמים התחלה של מילה והמערכת מחזירה לנו הצעות לחיפושים. התקשורת עם השרת מתבצעת ללא מעבר לדף הבא.

דוגמא נוספת, ציר הזמן בפייסבוק שטוען נתונים ברגע שהגענו לסוף ציר הזמן.

**יתרונות:**

* היתרון הגדול ביותר זה שהוא מרחיב את השימושיות של האתר (web application).
* חוסך שתעבורת הנתונים – טוען ומקבל בכל פעם רק מידע שדרוש ולא את כל הדף.
* לשיפור שיתוף של משאבים, עדיף לנצל את הכוח של כל מחשבי הלקוח, ולא רק שרת רשת ייחודי
* יכולה לשנות באופן סלקטיבי חלק של הדף מוצג על ידי הדפדפן, ולעדכן אותו ללא צורך לטעון מחדש את המסמך כולו עם כל התמונות, תפריטים, וכו'..

**חסרונות:**

* אין אפשרות לחזור אחורה (כפתור חזור לא יעזור).
* המידע שנמצא ב AJAX לדוגמא טוקבקים אינם יכולים להימצא ע"י מנוע חיפוש מפני שהם לא חלק מהדף.
* בגלל שהפעולות שביצענו לא משנות את הכתובת, אם נשלח את הכתובת של האתר למישהו הוא לא יקבל את המיקום המדויק שבו ביקרנו (עשינו שינויים בדף באמצעות לחיצות).
* בגלל שהמידע המתקבל הוא משרת, המשתמש ימתין לנתונים החדשים לכן חשוב להציג למשתמש להמתין לנתונים כדי שלא יחשוב שהכפתור לא עובד.
* **אם JavaScript אינו מופעל, AJAX לא יכול לעבוד**.
* השימוש הנרחב JavaScript בדף עלול להאט מעבדים ישנים.

חשוב להבין, AJAX הוא אובייקט המאפשר חבלת מידע מהשרת ללא ביצע טעינה מחדש לכל הדף. זהו בעצם שימוש א-סינכרוני בדף האינטרנט בשונה מדפי אינטרנט ]שוטים שהם סינכרוניים.

ה AJAX מעביר את הנתונים הנוספים מהשרת ללקוח כקובץ XML, שהיא שפה המזכירה את HTML (שפה של תגיות) שמטרתה היא העברת נתונים בין אפליקציות.

ישנם 2 שרתים עיקריים שעליהם עובדים בשימוש ב AJAX, הם נקראים אפצ'י וטומקאט (TomCat).

אפצ'י, הוא ארגון ללא מטרות רווח שמפתח תוכנות OPEN SOURCE.

*\*\* צריך להתקין בבית את הגרסה של TOMCAT כדי להפעיל את השרת. כדי לפתח יש לפתוח ב ECLIPSE פרויקט חדש שנקרא Dynamic Web Project.*

* כדי לכתוב ב AJAX ניתן להשתמש ב JS וניתן להשתמש ב JQUERY כדי לכתוב פחות. בספרייה JQUERY ישנם את רוב הפונקציות שנכתבו ל AJAX.
* הוספת ספריית JS הינה הכרחית לעבודה ב AJAX.

איך זה עובד?

* ישנו מנוע AJAX הנמצא לאחר דפדפן המשתמש, אשר שולח בקשת HTTP לשרת ומקבל חזרה נתוני XML , לאחר
* מכן מנוע ה - AJAX מתרגם לשפת HTML חזרה לתצוגה למשתמש

השימוש ב INNER HTML :

ע"י קיבוע אלמנט המאפיין - INNER HTML , האלמנט משנה את תכולתו.

התוכן הוא כל מה שכתוב בין שני תגים , בדר"כ שימוש בתגי DIV ו SPAN

התוכן החדש יכול להכיל שורת HTML

## תכונות AJAX

* שליחת תגובה משרת יכול להיות כל דבר, כולל: XML, דפי HTML נוספים, שברי HTML, - שימושי במיוחד! נתוני טקסט ...
* onreadystatechange - בחזרה קריאה לפונקציה עבור שינויים.
* readyState - המצב הנוכחי של השיחה (קריאת ה HTTP)
* HTTP. responseText - תוצאה הטקסט של הבקשה.
* responseXML - אובייקט XML DOM מן הבקשה. סטטוס - קוד מצב HTTP של התגובה.
* statusText - המצב. בטקסט מסוג HTTP
* סוגי HTTP States / AJAX Events:
  + כאשר נשלחת בקשה לשרת, אנו רוצים לבצע פעולות מסוימות בהתבסס של מצב התגובה מהשרת.
  + readyState הוא המשתנה המחזיק את הסטטוס של השיחה עם ה XMLHttpRequest בעל ערכים - 1-4
  + אירוע onreadystatechange מופעל בכל פעם את שינויי readyState
  + ערכי ה readyState: (xhttp.readyState):

1. – לא מאותחל – לפני שנפתחה קריאה
2. אתחול, אך לא נשלחו עדיין נתונים

2 - נשלח, אבל לא התקבלה תגובה עדיין

3 - נתונים חלקיים נשלחו.

4 – כל הנתונים נשלחו.

* + סטטוסים xhttp.status:
    - 200 – הכל תקין
    - 404 – העמוד לא נמצא.

אז מתי חיוג חוזר נקרא? מצבים 0, 1 אינם מדווחים על ידי דפדפנים רבים. מצבים 2, 3 מדווחים על ידי רוב הדפדפנים. מצב 4 מדווח תמיד.

**שליחת הבקשה:**

לשליחת הבקשה אנו משתמשים ב"פתח" ו"שלח" מהמתודות של אובייקט הבקשה, אך יש עוד עבודה לפני -

שהבקשה נשלחת...

דוגמא לשליחת בקשה:

http\_request.open(“GET”, “http://www.xyz.com/whatever”, true);

http\_request.send(null);

xhttp.open("GET", "ajax\_info.txt", true);  
xhttp.send();

**קבלת התוצאה:**

הבקשה נשלחת בצורה א סינכרונית, בצורה זו , דף ה- - - HTML עדיין מגיב וקוד ה- JavaScript ממשיך לרוץ בזמן שאובייקט הבקשה מחכה לתגובת השרת.

דוגמא לקבלת תגובה מ AJAX:

document.getElementById("demo").innerHTML = xhttp.responseText;

**קיבוע פונקציית callback :**

לפני שליחת הבקשה אנו צריכים לקבוע את אובייקט הבקשה, המאפיין onreadystatechange , זה נעשה בקלות ע"י שימוש תמיכה בפונקצית הקוד של JavaScript .

כאשר ישנה קריאה לCallBack :

הציפייה היא שפונקציית ה- - Callback תיקרא רק פעם אחת בקבלת התגובה, אך היא נקראת יותר מחמש

פעמים, ובכל פעם עם מצב HTTP שונה .

בתוך הפונקציה, מצא את המצב הנוכחי של ה- - HTTP בשימוש במאפיין אובייקט הבקשה readyState

דוגמה לפונקצית callback:

http\_request.onreadystatechange = **function() {**

**if (**http\_request.readyState != 4**) return;**

**if (**http\_request.status != 200**) {**

alret("Server reported error code: " + http\_request.status);

**return;}**

## Object Oriented on Java Script

* ניתן להשתמש ב this
* הכרזה על אובייקט מתבצעת בשני אופנים:
  1. var x = new Object();
  2. x = { } ;
* הוספת שדה חדש לאובייקט מתבצעת ע"י שימוש בנקודה:
  1. דוגמה: x.y=8 , אומר שנוסף ל X שדה חדש Y בעל ערך 8.
* הגדרת קונסטרקטור: ניתן להגדיר קונסטרקטור (בנאי).
* ירושה ופולימורפיזם: מתבצע ע"י שימוש במילה prototype .
  1. דוגמא: Tiger.prototype = new Cat (); - כאן הגדרנו שטייגר יורש מחתול.

# JSON – JavaScript Object Notation

JSON זה התקן להעברת נתונים בין אפליקציות. הוא למעשה מחליף של XML.

JSON מבוסס על JS והיא שפה קלה לכתיבה ולהבנה. השפה מסובבת על שפות ממשפחת C הכוללות את שפות C,C++,JAVA,PYTHON וכד'... וזה מה שעושה את השפה הזו אידיאלית להעברת נתונים בין אפליקציות.

גישה שנותנת להגדיר אובייקטים בגאווה סקריפט כמחרוזת.

חשוב מאוד למערכות מידע מודרניות.

משתמשים כתקן פופולארי ביותר להעברת נתונים באפליקציות שונות.

אף על פי שפורמט זה פותח בתחילה לשימוש ב-[Javascript](http://he.wikipedia.org/wiki/Javascript" \o "Javascript) ונעשה בו שימוש נרחב בשפה זו, JSON **נחשב כפורמט מבנה נתונים לא תלוי שפה, ותמיכה מוכנה בהמרה לפורמט JSON קיימת במגוון שפות תכנות.** הוא מבוסס על קבוצת משנה של JavaScript. הוא פורמט טקסט הוא שפה עצמאית לחלוטין, אך עושה שימוש במוסכמות, כי הם מוכרים ע"י מתכנתים של משפחת שפות C, כולל C, C + +, C #, Java, JavaScript, Perl, Python, ועוד. תכונות אלו הופכות את JSON לשפה האידיאלית לחילופי לנתונים.

## תכונות JSON

* מה קובע את התקן זה? אם יש לי סוגריים מסולסלים זה אובייקט.
* אם הם מרובעים זה מערך. - מערך הוא סוג של אובייקט.
* מערכים מסודרים ברצף של ערכים מערכים הם עטופים [] "," המפריד בין ערכי JSON לא מדבר על אינדקס. מערך יכול להתחיל מאינדקס 0 או 1.
* מקודד JSON חייב לקבל כל קובץ טקסטJSON שנעשה כמו שצריך
* מפענח JSON חייב לייצר קובץ טקסט חוקי שבנוי נכון
* מפענח JSON יכול גם לקבל את טקסט שלא נעשה ב.JSON
* JavaScript הוא קבוצת על של JSON.
* מהדר (Compiler) שלJavaScript הוא מפענח JSON.
* JSON הוא ייצוג פשוט, נפוץ של נתונים.
* תקשורת בין שרתים ולקוחות בדפדפן.
* שפה עצמאית חילופי נתונים ואפשרית לשימוש בשפות אחרות ושונות כמו PHP,C.

**דוגמא לקוד:**

# 

# JSP - Java Server Page

JSP היא טכנולוגיית צד-שרת מבוססת Java המאפשרת יצירה של דפי  HTML, XML וקבצים אחרים, בתשובה לבקשות  .HTTP טכנולוגיה זאת, מאפשרת לקוד Java ופעולות מוגדרות לשנות וליצור דפים בתשובה לכל בקשה.

התחביר של JSP כולל תגי XML נוספים, הנקראים פעולות JSP, המאפשרות לקרוא לפעולות של השפה. בנוסף, הטכנולוגיה מאפשרת יצירה של ספריות תגי JSP, שיכולות להשתלב עם תגי XML רגילים. ספריות התגים מאפשרות ליצור יישומי צד-שרת שאינן כבולות לשרת Web ומערכת הפעלה מסוימים.

דפי JSP מהודרים על ידי מהדר JSP לכדי Java Servlets . מהדר ה־ JSP יכול ליצור את ה־Servlets כבר בשפת הביניים של Java או בשפת Java ממש, שתהודר בתורה על ידי מהדר Java רגיל.

## Servlets:

טכנולוגית צד-שרת המבוססת JAVA ומהווה את המקבילה של ג'אווה לטכנולוגיות ליצירת אתר דינאמי כדוגמת ASP ו PHP. סוורלטים ניתנים ליצירה באופן אוטומטי מתוך JSP. ה Servlet בעצם מאפשר אינטרקציה עם לקוחות ברשת ויצירת אפליקציות מבוססות רשת, אפשר להסתכל על SERVLET כ הרחבה של שרתי וואב.

ההבדל בין Servlets ל-JSP הוא בכך שב-Servlets משבצים קוד HTML בתוך קוד Java , ואילו ב-JSP זה הפוך, שם קוד Java משובץ בתוך קוד HTML.

*Apache Tomcat – אפצ'י טומקט הוא שרת רשת ומריץ servlet.*

שימושים נפוצים של Servlets:

* עיבוד או אחסון נתונים שנשלחו (submitted) מתוך טופס HTML.
* לספק תוכן דינמי כדוגמת התוצאה של שאילתה על בסיס נתונים.
* ניהול מידע אודות המצב (state), שבאופן טבעי אינו קיים בפרוטוקול ה-HTTP שהוא כשלעצמו  stateless protocol. לדוגמה: שימור תוכן "עגלת הקניות" של מבקר באתר קניות באינטרנט, בזמן שהוא מדפדף בין דפי האתר.

**יתרונות ה Servlets:**

* יעילות: כל הסוורלטים רצים באותו תהליך.
* נוחות: ממשק API יחיד קל לשימוש ולתכנות.
* עוצמתי: עוזר לייצר אתר אינטרנט מורכב וטוב יותר בעל יכולות מתקדמות.
* ניידות – ניידות מלאה על פלטפורמות ושרתים שונים.
* אבטחה – מאפשר אבטחה טובה של אתר האינטרנט.
* עלות נמוכה – אפשרי לשימוש על שרתים חינמיים.

טכנולוגיות דומות נוספות:

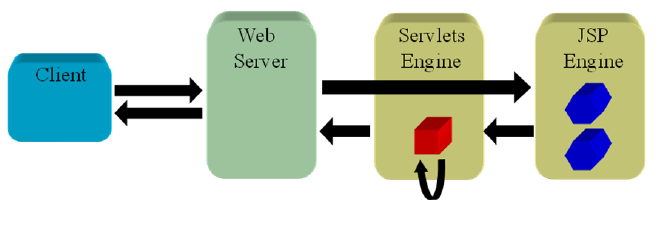
* PHP, ASP, CGI, ISAPI

JSP And Servlets: JSP ו Servlets ניתנים לשילוב באותה אפליקציה:

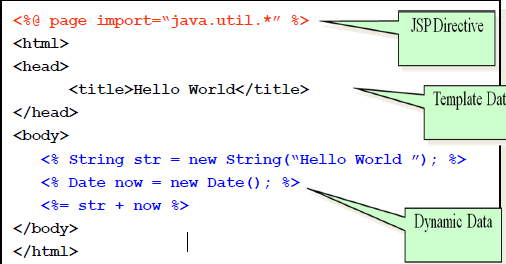
* JSP משמש ליצירת תצוגת תוכן דינאמית.
* Servlets משתמש ליצירת תוכן הדורש תכנות מורכב יותר.

**J2EE** פלטפורמת Java של חברת אורקל לפיתוח תוכנות enterpris . הפלטפורמה מספקת ממשקי תכנות יישומים (API) וסביבת הרצה (runtime environment) המשמשים לפיתוח והרצה של network services ,web services  ויישומי רשת מורכבים אחרים, הבנויים בארכיטקטורת שרת-לקוח רב-שכבתית, ומאופיינים בסילומיות (scalability), אמינות ואבטחת מידע.

Java Enterprise Edition מרחיבה את ה-Java Standard Edition, על ידי הוספת ממשקי תכנות יישומים התומכים במיפוי אובייקטי-רלציוני (עבודה עם בסיסי נתונים), ארכיטקטורות מבוזרות ורב-שכבתיות, web services ועוד. הפלטפורמה מבוססת על רכיבי תוכנה מודולריים אשר רצים על גבי שרת יישומים. תוכנה עבור Java EE מפותחת בעיקר בשפת התכנות Java ומשתמשת ב-XML לצורכי קונפיגורציה.

****

## JSP Syntax



# Cookies and Session Management

קוקיס זהו קובץ קטן אותו מעביר השרת ללקוח המכיל מספר ארוך המהווה מספר מזהה של מחשב ודפדפן. באמצעות קובץ זה השרת יכול לדעת לשמור את הנתונים ללקוח הספציפי, ויכול לעזור בניהול ה session אשר שומרים נתונים עם הלקוח.

דוגמא, אתר קניות. המוצרים שבחרתי לרכוש נשמרים בעגלה בכל זמן שאני מטייל באתר. לשרת יש אפשרות להגדיר תוקף לקוקיס, ובכך לבטל את הקוקיס אחרי זמן מסוים. לדוגמא, העגלה מתרוקנת לאחר יום.

Session נפתח:

* בפנייה של מחשב לאפליקציה.

Session נסגר:

* לאחר סיום תוקף (time-out).
* באמצעות מנגנון אשר סוגר את ה session לדוגמא (Log Off).

**cookies**

cookies הינם פיסות מידע טקסטואלי ששרת ה-Web שולח לדפדפן והוא מחזירם ללא שינוי בעת הביקור הבא באותו אתר. באמצעותם ניתן לאחסן מידע אודות session וכל קשר הנובע מכך יכול לחפש את ה-session הנוכחי ולחלץ מידע אודות ה-session ממקום מסוים על מחשב השרת. לאחר מכן בדרישות נוספות השרת יכול להשתמש ב-hash table שהוא מתחזק כדי לקשר בין session id ל-cookie המתאים. זוהי הגישה הנפוצה לטיפול  ב-sessions אולם קיימים מספר חסרונות לשיטה זו: חלק מהמשתמשים מבטלים את אפשרות השימוש ב-cookies לשם הגנת הפרטיות. חסרון נוסף הוא שהתחזוקה והמעקב על cookies נעשים ברמת ממשק נמוכה.

* קוקיס הינו חלק מפרוטוקול HTTP.

חסרונות השימוש ב Cookies:

* דורש המון עבודה מצד המתכנת
* המידע שמאוכסן הוא טקסטואלי בלבד
* אבטחת המידע אינה טובה – המידע עובר בין השרת ללקוח.
* לקוחות תומכים ב 20 קוקיס לשרת ועד 300 בכלל.
* Cookies מוגלבים ל 4kb לכל אחד.

## Session Management

בפניה חדשה לשרת נפתח session שמקבל id משלו, בכך בפניות הבאות השרת יוכל לזהות את הפניה ולשייך אותה למשתמש שפנה. בתוך jsp יהיה אובייקט session שישמור את המזהה הזה עבור כל משתמש בנפרד,ובכך יהיה ניתן להתייחס אל המשתמשים בצורה שונה. יש מנגנון timeout או logout לסגירת ה session.

**מעקב sessions  ב-Servlets**

Servlets מספקים ממשק ברמה גבוהה למעקב אחרי sessions הבנויים על גבי cookies או שכתוב URL. למעשה רוב השרתים משתמשים ב-cookies במידה והדפדפן תומך בהם ומשנים אוטומטית את מעקב   
ה-sessions לשיטת שכתוב URL אם ה-cookies אינם נתמכים או שבוטלו במפורש. מפתח ה-Servlet אינו צריך להטריד עצמו בטיפול בפרטים ואינו צריך להפעיל ישירות cookies או להוסיף מידע לכתובת ה-URL אלא הוא מקבל מקום נוח לאחסון אובייקטי תיווך הקשורים ל-sessions.

בכל חיבור בין השרת ללקוח, נשלח SessionID המזהה את הלקוח באופן חד ערכי. ישנם 3 דרכים עיקריות למתן ID ייחודי:

* 1. באמצעות cookies
  2. באמצעות URL
  3. באמצעות שימוש ב SSL.

מתי משתמשים ב SESSION?

* אפליקציה שיש בה LOGIN
* אפליקציות שבהן יש תהליך בהמשכים

מאפיינים נוספים:

* אם משתמשים ב HTTPS לא צריך קוקיס.
* ניתן להשתמש בהעברת כתובת URL שנושאת את הפרטים החשובים ללא שימוש בקוקיס.

שאלות למבחן?

* מתי SESSION נוצר? כאשר אנ פונים לכל משאב שנמצא בתוך האפליקציה
* ביצעת התחברות לאתר ויצאת מהדף, לאחר מכן הגיע מישהו אחר והתחבר לדף מאותו דפדפן. האם הוא יוכל לעבוד על החשבון שלך? **כן**
* האם לכל SESSION יש TIMEOUT קבוע? לא! בהתאם לאפליקציה

**HTTPS – HTTPSession – מאפשר ל JSP לבצע:**

* אכסון של אובייקטים בתוך ה SESSION. ושימוש של אותו אובייקט מס פעמים באותה בקשה.
* לקבל מידע על ה SESSION כמו מס' מזהה, זמן יצירה, זמן חיבור אחרון וכו'..
* שימוש מידע כגון Session Timeout.

**נקודות חשובות:**

* Cookies משמשות לשמירה על מידע.
* שימוש ב SESSION עוזר להתמודד עם הבעיות של COOKIES ומאפשר עבודה עם אובייקטי JAVA.
* Session Management משמש בכל האפליקציות הבסיסיות.

## https://upload.wikimedia.org/wikipedia/he/thumb/e/e2/Three_tier_architecture_heb.png/800px-Three_tier_architecture_heb.pngServer Side

**Business Logic**

היא חלק ממערכת תוכנה שתפקידו לממש את הכללים העסקיים מ"העולם האמיתי". כללים אלו קובעים כיצד נתונים יכולים: להיווצר, להשתנות, להיות מוצגים, ומאוחסנים. לרוב, המושג מתאר את החלק בתוכנה אשר מכיל את האלגוריתמים הפונקציונליים שמטפלים בהעברת מידע בין בסיס נתונים לממשק משתמש.

נהוג להבחין בין הלוגיקה העסקית לבין יתר חלקי התוכנה שיכולים להוות את התשתיות, או שהם אחראים על פרטים כגון ניהול בסיס נתונים, הצגת ממשק משתמש, והחיבור הכללי בין חלקי המערכת השונים.

## JavaBeans

[קישור לנושא באתר ויקיפדיה (כולל דוגמה)](https://he.wikipedia.org/wiki/JavaBeans)

הם רכיבי תוכנה הניתנים לשימוש חוזר אלו הן מחלקות הנכתבות בשפת ג'אווה ומשמשות כדי לכמס הרבה אובייקטים בתוך אובייקט אחד (ה-BEAN) כך שניתן יהיה להעביר אותם ממקום למקום כאובייקט bean אחד, במקום הרבה אובייקטים בודדים.

באמצעות USEBEAN ניתן לקבל נתונים מטפסים בצורה יותר קלה.

שאלה: האם אפשר להחליף UseBean בקוד ג'אווה זהה? **תשובה:** כן! המטרה שלל javaBeans הוא לקצר את כתיבת הקוד.

**scope:**

<http://webmaster.org.il/articles/jsp-orders1>

על פי הגדרת ה SCOPE ייקבע אופן השימוש בנתונים. לדוגמא, האם הנתונים יהיו שונים לכל משתמש או זהים לכלל המשתמשים דוגמא, כשאנו נכנסים לאתר ומבצעים התחברות, אנו רואים את שמינו מוצג באתר השם הוא שונה לכל משתמש ולכן משתמשים ב scope-session.

סוגי SCOPE:

Request – רק לבקשה זאת עד משלוח תשובה.

Session – תחום ההכרה הוא של המשתמש הנוכחי. לפי משתמש עד זמן ביטול הSESSION עבור כל הדפים באפליקציה. מספר האובייקטים הנוצרים כמספר ה SESSION

Page - תחום ההכרה הוא בעמוד הנוכחי בלבד.

Application – נגיש לכל האפליקציה (ולכל המשתמשים).

**JSP And JavaBeans:**

ישנם 3 תגיות שאיתם משתמשים ב JSP לשימוש ב JB:

* + **<jsp:useBean>**
  + **<jsp:setProperty>**
  + **<jsp:getProperty>**

דוגמה:



# שאלות

1. מה נכון לגבי HTML ?
2. **תג <a> משמש ליצירת קשר לדף HTML אחר .**
3. HTML זה סוג של שפת תכנות.
4. **לתווי רווח רבים יש השפעה ויזואלית זהה לתו רווח אחד.**
5. **כל** **תג פתיחה חייב בתג הסגירה המתאים.**
6. **בתג <a> אפשר להשתמש ליצירת קשר לחלקים שונים במסמך HTML**
7. HTML
8. **<br> בונה שורה ריקה גם אם יש שורת טקסט אחת במסמך.**
9. HTML מבוסס על XML
10. **לכל TAG יכול להיות ATTRIBUTE**
11. **ע"י TAG נקבעת התצודה בדפדפן**
12. **מסמך HTML הוא מסמך שמכיל טקסט בלבד.**
13. מה נכון לגבי אלמנט DIV ?
14. **אלמנט DIV בונה פסקה**.
15. **אלמנט DIV אינו כולל עיצוב מיוחד כברירת מחדל**
16. **תוכן של אלמנט DIV אפשר לשנות בצורה דינאמית ע"י שינוי מאפיין INNERHTML**
17. לאלמנט DIV יש תפקיד שונה מתפקידו של אלמנט div .
18. במסמך HTML יכול להיות אלמנט DIV אחד בלבד .
19. מה נכון לגבי אלמנט FORM במסמך HTML ?
20. **באלמנט FORM אחד יכולים להיות מס' כפתורים מסוג SUBMIT**
21. **אלמנטים INPUT מיועדים להיות בתוך אלמנט FORM .**
22. **לכל אלמנט INPUT נותנים שם לפיו מעבדים נתונים המתקבלים מטופס בשרת .**
23. קליק על כפתור SUBMIT גורם לשליחת נתוני הטופס לשרת והעברה לדף HTML אחר.
24. במסמך HTML אחד יכול להיות אלמנט FORM אחד בלבד .
25. מה נכון בנוגע למשתנים ב JAVASCRIPT ?
26. **אם המשתנה מוגדר מחוץ לפונקציה תחום הכרתו כל המסמך כולל חלק שבתוך אלמנט BODY.**
27. משתנים אפשר להגדיר רק בתוך חלק של הקוד שנמצא באלמנט HEAD של מסמך HTML .
28. למשתנה שמוגדר בתוך פונקציה ללא שימוש בהגדרה var יש גישה גם מחוץ לפונקציה
29. למשתנים אין סוג נתונים
30. **למשתנה שקיבל ערך מספרי אפשר בהמשך לתת ערך של מחרוזת.**
31. מה נכון בנוגע לתכנות מונחה עצמים ב – JAVASCRIPT ?
32. **לבניית אובייקטים משתמשים באופרטור new**
33. אי אפשר להגדיר בנאי .
34. **להגדרת מחלקה לא משתמשים במילה השמורה class .**
35. **שימוש ב- json מאפשר דרך להגדרת קבוצת נתונים בצורת אובייקט .**
36. **גישה לנתונים חברי מחלקה אפשר לקבל גם באמצעות אופרטור [ ] .**

1. מה נכון בנוגע ל- AJAX ?
2. מגדירים פונקצית תגובה שמנגנון AJAX הפעיל אותה רק פעם אחת בעת קבלת נתונים מהשרת .
3. **מנגנון AJAX מאפשר תצוגה בדפדפן נתונים נוספים משרת תוך המשך עבודה במסמך המוצג .**
4. כפתור back של דפדפן מאפשר לחזור לנתונים שהיו מוצגים לפני הפעלת מנגנון של AJAX .
5. **AJAX מאפשר לקבל נתונים משרת ללא קבלת מסמך HTML כולו .**
6. נתונים שמקבלים דרך AJAX חייבים להיות בצורת מסמך XML .
7. מה נכון בנוגע למערכים ב- JAVASCRIPT ?
8. **משתמשים בסוגריים מרובעים [ ] לגישה לאלמנטים של מערך.**
9. **להתחלת מערך משתמשים ב- new Array () .**
10. **למערך יש תכונה בשם length .**
11. במערכים אינדקס מתחיל מ- 0 וחייב להשתמש עד למספר ברצף מקסימלי.
12. כל הערכים במערך חייבים להיות באותו סוג.
13. מה נכון לגבי שורת הקוד הבאה :

Document.getElementById (a).style.backgroundColor = "red"

1. **השורה תשנה צבע רקע של אלמנט אחד בלבד.**
2. **תכונה style קיימת לכל אלמנט ויזואלי של מסמך HTML .**
3. **Document זה אובייקט הקיים בכל מסמך HTML.**
4. במקום הפונקציה getElementById(a) אפשר לכתוב getelementbyid(a)
5. I **a חייב להיות משתנה שערכו מוגדר לפני שורת הקוד .**
6. מה נכון לגבי DHTML ו- DOM ?
7. **במאפיין id של אלמנט משתמשים לקבלת גישה לאלמנט כאובייקט ב- javascript .**
8. 2 מילים של טקסט במסמך של HTML הן 2 אובייקטים ב-DOM .
9. DHTML זה שפה נוספת לHTML.
10. **לכל אלמנט של HTML אפשר לקבל גישה כאובייקט ב- javascript .**
11. **כל אלמנט של HTML הוא אובייקט ב-DOM , אפילו אם לא מוגדר לו id .**
12. מה נכון בנוגע למשפט הבא בקובץ JSP ?

<%=a.toString() %>

1. המשפט שגוי מפני שאין ; בסוף .
2. אפשר להחליף את המשפט בביטוי זהה ב- javascript .
3. המשפט יכול להיות ממוקם בקובץ רק אחרי תוית <body> .
4. **משתנה a חייב להיות מוגדר היטב לפני המשפט .**
5. **המשפט יוצר מחרוזת במסמך שנשלח למשתמש הקצה.**
6. מה נכון בנוגע לשימוש של JSP ו- SERVLET ?
7. שרת התומך בSERVLET חייב לתמוך ב JSP .
8. אפשר לפתח יישום web שמשתמש ב JSP ללא תרגום קבצי JSP .
9. **אפשר לפתח יישום web שמשתמש ב SERVLET ללא שימוש ב JSP .**
10. **שרת התומך ב JSP חייב לתמוך ב SERVLET .**
11. <a href=http://…/> link here</a>
12. **אלמנט A משמש ליצירת קשר עם מסמכים אחרים.**
13. **ערך HREF יכול להיות שם קובץ**
14. בשורת קוד יש שגיאה כי בURL אין שם קובץ
15. נהוג להגדיר HREF באבסולוט פאת'
16. **באלמנט A ניתן להשתמש כדי לקבוע קשר בין חלקים באותו המסמך**.

DIV:

1. DIV שונה מdiv.
2. **אפשר לשנות תוך DIV בצורה דינמית תוך שימוש בINNERHTML**
3. **DIV אינו כולל עיצוב ברירת מחדל**
4. בHTML יכול להיות DIV אחד בלבד.
5. **אלמנט DIV בונה פסקה**

**HTML:**

1. **דפדפן לא מבצע פעולות לאלמנטים שהוא לא מכיר**
2. **יכול להיות אלמנט ללא מאפיינים.**
3. שם האלמנט מסביר משמעות נתונים שבתוכו
4. שם אלמנט יכול להיות כל מחרוזת שהיא.
5. דפדפן אינו מציג תוך אלמנטים שהוא אינו מכיר.

CSS:

1. **קיימים סגנונות שיש להם מראה שונה בדפדפנים שונים**
2. **אפשר להגדיר CSS בקובץ נפרד**
3. **אפשר לקבוע מיקום בדפדפן עפ"י CSS**
4. COLOR מוגדר בSTYLE וב ATTRIBUTE שניהם תקפים (גל סימן)
5. אין דרך להגדיר סגנון לאות בודדת במילה גם בשימוש אלמנטים נוספים ב HTML

JS:

1. פונקציות אינן יכולות להחזיר ערך
2. **אפשר להגדיר משתנים מקומיים שמוכרים בתוך פונ' בלבד**
3. **אפשר להגדיר פונקציות בקובץ נפרד עם סיומת JS**
4. **פונ' יכולות לקבל פרמטרים**
5. בתוך פונקציה חייב להיות RETURN

>IMG SRC...onclick="do.this;do.that"

1. ONCLICK חייב להיות באותיות קטנות
2. **אפשר לגשת לאלמנט דרך JS**
3. ONCLICK חייב לקבל שם של פונ'
4. אי אפשר לשנות תמונה דרך JS
5. **פונ' בשם DO.THIS / DO.THAT** חייבות להיות מוגדרות ב JS

JQUERY

1. **בנויה כך שאין צורך בקוד שונה לדפדפן שונה**
2. נוצר כדי לקשר בין JS ל JAVA
3. פונקציה דולר מחזירה מערך אובייקטים
4. **לשימוש בJQ חייב להגדיר קשר לקובץ JS שמכיל קוד של JQ**
5. **מטרת JQ היא לאפשר כתיבת מינימום קוד לביצוע איגודים טיפוסיים עבור קבוצות אלמנטים של מסמכי HTML**

**JSON**

1. **להגדרת אובייקט משתמשים ב {}**
2. בביטוי אחד של JSON אי אפשר להגדיר מערך אובייקטים
3. שם הנתון חייב להיות ללא גרשיים
4. JSON הוא סוג של שפת תכנות
5. **ביטוי בצורת JSON יכול להיות בתוך קוד של JS**

מה נכון בנוגע ל- **AJAX** ?

1. מגדירים פונקצית תגובה שמנגנון AJAX הפעיל אותה רק פעם אחת בעת קבלת נתונים מהשרת .
2. **מנגנון AJAX מאפשר תצוגה בדפדפן נתונים נוספים משרת תוך המשך עבודה במסמך המוצג .**
3. כפתור back של דפדפן מאפשר לחזור לנתונים שהיו מוצגים לפני הפעלת מנגנון של AJAX .
4. **AJAX מאפשר לקבל נתונים משרת ללא קבלת מסמך HTML כולו .**
5. **נתונים שמקבלים דרך AJAX יכולים להיות בצורת מסמך json .**

**שאלות נוספות:**

* כמה form יכולים להיות על דף HTML אחד? כמה שרוצים
* האם בform יכול להיות יותר מ-submit? כן
* איפה אפשר לכתוב CSS? בדף CSS נפרד, body\head או באובייקט עצמו.
* איזה שפות משותפות לכל הבראווז'רים? HTML וJS
* מה ההבדל בין טקסט להיפר-טקסט
* מה ההבדל בין HTML לcss ? HTML- הצגת הנתונים על המסך. CSS – עיצוב הנתונים על המסך.
* מה היא השפת תיכנות היחידה שעובדת בצד לקוח? JS
* מה קורה אם יש שגיאה בקובץ HTML? לא קורה כלום, זוהיא סביבה סלחנית.
* מה זה אלמנט? כל מה שנמצא בתוך הפקודה.
* מה כולל אלמנט? תג פתיחה, מלל\פקודות, תג סגירה
* מה ההבדל בין head ל body ? head – מידע איך להציג את המידע. Body – המידע עצמו.
* מה ייקרה אם נרשום שני body? הוא יאחד אותם.
* האם יכול להיות מאפיין בתג סגירה? לא, רק בתג פתיחה.
* האם אנטר משפיע על תצוגה HTML? לא
* מה ההבדל בCSS בין נקודה לרישתיות? # - id , נקודה – class
* האם לכל מאפיין יש שם וערך? כן.
* האם צריך לרשום את שם הטיפוס בJS? לא
* מה עושה ALT ? תיאור טקסטואלי של התמונה במקרה שהתמונה לא עולה....
* בקשר של טבלאות – התא הגבוהה ביותר – קובע את הגובה של הטבלה והרוחב הרחב ביותר קובע את הרוחב שלה.
* האם מסמך HTML כולל תמונות? לא. הוא רק מפנה לנתיב של תמונה.
* מה ההבדל בין get לpost? Get שולף את הנתונים ומוסיף אותם לurl ופוסט מסתיר את הנתונים ולא מוסיף אותם לURL
* מה הסיבה להשתמש בget? ע"מ לדבר ושכמות הנתונים קטנה.
* מה היא הדרך לקבל מידע מהמשתמש? רק דרך טפסים.
* איפה רושמים סקריפטים במסמך HTML? איפה שרוצים.
* האם למשתנים של JS אין סוג? לא נכון – יש לו סוג אך הוא נקבע אחרי בניית המשתנה.
* איך מקשרים בין פעולה בJS לHTML? ע"י כך שנותנים לאלמנט ID\CLASS
* מה זה INNERHTML? אפשרות לייבא אובייקט חדש למסך שלא הופיע קודם.
* מה השם המלא של JS? ECMASCRIPT
* האם מותר להתחיל שם של מזהה במספר? לא, רק עם אותיות (אפשר גם: \_ , $)
* JQ – האם פונקציית $ מחזירה מערך אלמנטים? לא. היא מחזירה אובייקט שעוטף את האלמנטים.
* מה הוא Nan? לא מספר. למשל חלוקה באפס או חלוקה של מספר בstring
* האם Nan = Nan? לא!
* בפונקציית בוליאנית – אילו ערכים יחזירו ערכי שקר? 0 , NAN, " " , null, false, undefined
* מה מסמל הסימן === ? השוואה של שני דברים מבלי להמיר את הצדדים לבוליאנים.
* איך להגדיר ירושה בJS? א. P1=object (p) ; ב. p1=prototype p (p1 יורש מ- P)
* האם ניתן לשים פונקציה בתוך קונסטרקטור בJS? כן
* מהו web2? בweb רגיל כל אחד יכול להעלות אותו אך מתקשה לתחזק. Web2 – בעל האתר מעלה אותו אך מישהו אחר מתחזק אותו. כגון: ויקיפדיה ואתרי צ'טים.
* מה זה AJAX בשתי מילים? יכולת באמצעות אובייקט אחד לשלוח שאילתא ולקבל תשובה.
* ע"מ להחזיר אתר web2 חייבים? לעשות שימוש בAJAX וג'ייסון.
* האם יכול להיות פרוייקט שמשתמש רק בJSP ולא בסרבלט? לא
* האם יכול להיות פרוייט שמשתמש רק בסרבלט ולא בJSP? כן!
* מה הפירוש של "נוזל שמשות"? (שאלה בחסות רמי אביטן- את התשובה תמצאו אצלו..)
* לאן יהיה מתורגם הקובץ JSP? למתודת SERVISE במחלקת סרבלט
* <int x ; %!%> זהו משתנה חבר מחלקה? כן. (ובלי הסימן קריאה הוא לוקאלי)
* האם מותר לרשום למשתנה לוקאלי public? לא!
* מה הוא web Application? אפליקציה שהUI שלה מוצג בבראוז'ר.
* מה היא התיקייה שחייבת להיות בכל web Application ? web-inf
* מתי אובייקט סשן נולד? כאשר יש פנייה ראשונה לשרת
* מתי אובייקט סשן נסגר? כאשר עושים ניתוק או timeout
* איפה נשמרים העוגיות? בהארד דיסק.
* כמה מאפיינים יכולים להיות לאלמנט? בלי סוף, לא חייב להיות מאפיין לאלמנט.
* האם id חייב להיות ייחודי ? כן!
* מהם שלושת הדברים החשובים באפליקציה? UI , BL , DB
* איזה חלק מהאפליקציה רץ בצד שרת? BL +DB
* ממה מורכב מאפיין? משם המאפיין והערך שלו.
* בJSP נשלח קוד למשתמש הקצה? לא
* אם הגדרתי משתנה בJSP – האם הוא מוכר בעמודים אחרים? לא נכון
* האם JS היא מונחת עצמים? כן
* האם יש בJS קונסטרקטור? כן. אבל אין את המילה class
* כמה דרכים יש לפתיחת אובייקט בJS ? 3
* מהם הדרכים שבאמצעותם ניתן לשנות אלמנטים בHTML? JS (innerhtml) – STYLE
* האם יש משו שאני יכולה לעשות רק בJQ ? לא , הכל אני יכולה לעשות בJS
* האם העיצוב שלי נראה אותו דבר בכל הבראווז'רים? לא. אך ניתן לאכוף את זה אם מתכנתים בJQ .