**תוכן עניינים**

**HTML**

תכונות.....................................................................................3

מאפייני אלמנטים.......................................................................3

.................................................................4HTML terminology

..........................................................4.HTML Document tags

.........................................................................5Heading tags

.......................................................................5Paragraph tag

טבלאות..................................................................................6

FRAMES..............................................................................7

FORM..................................................................................7

INPUT..................................................................................7

תציב יחסי ואבסולוטי................................................................8

שאלות ותשובות חזרה HTML...................................................8

**CSS**

תכונות...................................................................................9

PADDING............................................................................9

MARGING...........................................................................9

BORDER.............................................................................9

**JAVASCRIPT**

תכונות..................................................................................10

HTML AND JAVASCRIPT.................................................12

משתנים...............................................................................13

פונקציות..............................................................................13

אירועים...............................................................................14

אובייקטים............................................................................15

מערכים...............................................................................15

**DHTML**

תכונות............................................................................16

DIV...............................................................................16

SPAN...........................................................................17

תכניקות.........................................................................17

DOM............................................................................18

**AJAX**...........................................................................19

**JSON**...........................................................................23

**JQUERY**......................................................................25

**נספחים**.........................................................................27

**HTML**

**תכונות:**

* 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.

**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>**

**HTML terminology**

* תגיות הן פקודות HTML
* כל תגית מתחילה ב< ונגמרת ב> לדוגמא <HTML>
* תג הסגירה ייראה כך </HTML>
* חלק מהתגיות לא דורשים תג סוגר <BR>
* תכונות הן קודים מיוחדים המופיעים עם תג ההתחלה.
* התכונות מגדירות את התגית הקשורה <body bgcolor=“red”>
* ישנם שלושה חלקים ב יסוד מסמך HTML :
* מתחיל בתגית עם התכונות
* תוכן
* תגית סיום \ סגירה

**HTML Document tags**

* **<Html>…</Html>**
* אלמנט השורש של המסמך כולו.
* ממוקם בשורה הראשונה והאחרונה של המסמך.
* **<Head>…</Head>**
* מספק מידע תיאורי על מסמך בחלק הכותרת-header..
* **<Title>…</Title>**
* מוצג בחלק העליון של המסמך
* ממוקם בתוך תגית ה <Head> tag
* **<Body>…</Body>**
* מזהה תוכן של דף האינטרנט
* רב התגיות נמצאות בתגית הנל

**Heading tags**

* **<H1>…</H1>**
* כותרות מוגדרות עם <h1> לבין <h6> תגים.
* <h1> מגדירה את הכותרת הגדולה.
* <h6> מגדירה את הכותרת הקטנה ביותר.
* Attribute: align (left, center, right).

**Paragraph tag**

* **<P>…</P>**
* מזהה טקסט בתוך דף.
* Attribute: align (left, center, right).

**טבלאות**

<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.

**FRAMES**

* 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.

**FORMS**

* 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.

**נתיב יחסי ואבסולוטי ההבדלים**

* נתיב יחסי ונתיב אבסולוט. אם רשום HTTP: אז זה כתובת.
* אם שמים רק את השם של הקובץ או התיקייה שלו אז זה נתיב יחסי.
* **אסור לעשות נתיב שנמצא בתוך המחשב שלנו כי אז לא יהיה נגיש לאחרים אז רצוי רק כתובת אינטרנט או שם קובץ.**

##### ההבדלים - :Relative Paths index.html /graphics/image.png /help/articles/how-do-i-set-up-a-webpage.html :Absolute Paths <http://www.mysite.com/> <http://www.mysite.com/graphics/image.png> <http://www.mysite.com/help/articles/how-do-i-set-up-a-webpage.html>

* נתיב מלא (c:) !

“c://pic/1.gif” – נקרא full qualified path

אבל הנתיב משתנה ולכן זאת טעות

הנכון לכתוב רק נתיב יחסי Relative Path, למשל: 1.gif – זה הכי שימושי

מותר לכתוב URL (<http://www>)

אם אני לא רוצה שהתמונות יהיו באותה הספרייה מותר לי למשל pics/1.gif , זה עדיין במסגרת נתיב יחסי.

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**

**תכונות**

* 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)

**JavaScript**

**תכונות**

* גאווה סקריפט היא שפת תכנות
* **ג'אווה סקריפט מונחה עצמים!!!**
* 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 Mode**l אובייקטים של DOM המתארים מסמך מורכב ומגדיר גישה לאותם אובייקטים JS.

אובייקטים חשובים: החלון. המסמך. מיקום. ההיסטוריה

* זו שפה אמתית.
* קטנה, אבל מתוחכמת
* 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.
* ליצור אובייקט חדש - לשלושת הביטויים האלה יש בדיוק את אותה תוצאה:

**new Object()**

**{}**

**object(Object.prototype)**

**{}**

**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 גאווה 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.
* המשתנה יכול להכיל סוגים שונים בזמנים שונים. מספרים כולל שני מספרים שלמים ושברים. בוליאני. מחרוזות, NULL מערכים ברורים. לא מוגדר
* יש רק סוג מספרים אחד, אין שלמים רק IEEE-754 המוכר כ"דאבל"
* 0.1 + 0.2 = 0.30000000000000004

NaN - מספר מיוחד: לא מספר תוצאה של פעילות מוגדרת או שגויה Toxic: כל פעולה בחשבון עם NaN כקלט יהיה NaN כתוצאה NaN אינו שווה דבר, כולל NaN

**פונקציות**

* מילת מפתח FUNCTION.
* יכולים להיות פרמטרים.

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

* + אם נקח 4 פרמטרים הוא יקח את השניים הראשונים.
  + אפשר גם בלי פרמטר. והברירת מחדל 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.

ןid\_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 , לדוגמה :
* **function fade(id) {**

**var dom = document.getElementById(id),**

**level = 1;**

**function step () {**

**var h = level.toString(16);**

**dom.style.backgroundColor =**

**'#FFFF' + h + h;**

**if (level < 15) {**

**level += 1;**

**setTimeout(step, 100);**

**}**

**}**

**setTimeout(step, 100);**

**}**

* 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***]**

**אירועים כמו Onclick**

* באירועים כמו אונקליק וכו'... שמה עובדים רק גאווה סקריפט ואי אפשר להשתמש ב HTML

<a onclick=”only jave script”>

* **Onclick אפשר לכתוב עם אות גדולה אבל בגאווה לא! בגלל הרגישות.**
* אובייקטים רבים ליצירת "אירועים" לדוגמא onclick. אירועים מתרחשים בדרך כלל כתוצאה של פעולת משתמש לחיצה על קישורים או אלמנטים הטופס. טעינת מסמך חדש. וכו '
* "המטפל באירועים" הוא קוד מסוים אשר מבוצע כאשר אירוע מסוים מתרחש באובייקט מסוים ע"י לחיצה של המשתמש במעבר עכבר. וכו' "המטפל באירועים" יכול לבצע כל קוד JavaScript. הגדרת "המטפל באירועים" ע"י הוספת תכונה onEvent = "קוד" לתג של האובייקט. onFocus- אובייקט מקבל את הפוקוס קלט. onBlur- האובייקט מאבד את הפוקוס קלט. OnChange -עריכת שולט בלבד. האובייקט מאבד את הפוקוס קלט ותוכנו השתנו.
* **<A HREF="newdoc.htm" onClick="MyEventHandler()">**

**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.**

**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.

**DHTML basics**

* לכל אלמנט של HTML אפשר לקבל גישה.
* לא טכנולוגיה חדשה אלא טכניקה של שימור בין JS,HTML,CSS
* הוא נותן לגאווה סקריפט גישה ברורה.
* אין שפה מיוחדת או טכנולוגיה המכונה dhtml
* יש לי גישה לכל הנתונים כולל העיצובים.
* הוא לא קיים בנפרד הוא רק רעיון.
* <div> תמיד בונה פסקה חדשה ואפשר להשים <div> רבים ואחרים
* <div> ו<span> יש innrtHTML
* <div> לא קובע עיצוב רק פסקה חדשה.
* <span> מיועד להמשך לשורה נקודתי ולא פסקה שלמה.
* <span> אפשר לשנות מילה אחת או משהו בשורה או צבע או גדול אז אני אשתמש ב<span> .
* הם די דומים <div> ו<span> יחסית...
* <div> ו<span> משנים לא רק שינוי של עיצוב אלא אפשר גם לשנות תוכן שנמצא בדף HTML.
* <Span> ו <div> אין ברירת מחדל לא יראו כלום אם לא מעצבים אותם.
* מה שיהיה רשום ב<div> זה הכותרת -Id(div).innerHTML= <H1>FASF</H1>
* Dom שונה מHTML בכך שהוא עושה את אותה פעולה כמו <div> רק על כל הדף.
* אין שפה טכנולוגיה מיוחדת או בשם DHTML. Dynamic HTML, או DHTML הוא שילוב של HTML, של צד הלקוח שפת סקריפט JavaScript, הגדרת המצגת שפה (גיליונות סגנון מדורגים, CSS), ומודל אובייקט מסמך. DHTML משמש לעתים קרובות כדי לעשות לחצני גלגול או תפריטים שנפתחים בדף האינטרנט, השימוש הנפוץ פחות היא ליצור דפי דפדפן למשחקי פעולה.
* **תגיות <div> ו <span> ו innerHTML לגישה ישירה על כל הגופים הנראים ובלתי נראים למסמך HTML על ידי תוכנית JavaScript**.
* שינוי סגנונות SCC באופן דינמי על ידי תוכנית JavaScript.
* ההבדלים בין תגיות <div> ו <span> שמציגים את מקטע ה-div בשורה חדשה (הוספת שורה חדשה לפניו ואחריו) <span> לעומת זאת מוגדר כמקטע בתוך השורה בה הוא נכתב.
* חשוב לומר שמלבד הוספת מעברי השורה אין כל השפעה עיצובית לתג < <div.

## [div](http://he.wikibooks.org/wiki/XHTML10STRICT/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](http://he.wikibooks.org/wiki/XHTML10STRICT/span)

* אלמנט מסוג שורה.
* תוכן מסוג טקסט או שורה.
* מאפיינים כלליים (ליבה, שפה ואירוע)
* תגית זו מגדירה חלק מאלמנט על מנת לעצבו.
* . הבדל נוסף בין התגים הוא ש-div הוא אלמנט גוש ו-span הוא אלמנט שורה.
* Html דינמי או dhtml הוא שילוב של שפת סימון סטטי(html), שפה בצד הלקוח (javascript), הצגת הגדרות השפה (גיליונות סגנון מדורגים, css) ומודל אובייקטי הטופס.
* Dhtml משמש לעיתים קרובות כדי להפוך כפתורים או תפריטים הנפתחים בדף האינטרנט. השימוש הפחות נפוץ הוא ליצור דפדפן מבוסס משחקי פעולה.

טכניקות ראשיות ב dhtml-

* תגיות <div>, <span> והדברים הפנימיים השייכים להם ב html
* גישה ישירה אל כל החפצים הנראים והבלתי נראים של אובייקטי מסמך ה html ע"י javascript
* שינוי סגנונות scc בצורה דינאמית ע"י javaScript
* בחינה אינטנסיבית של מודל האירועים של DOM
* Dhtml- היכולת לבנות דף ורמת שינוי בלתי מוגבלת, אפשר להחליף תמונה על הטקסט.
* **Dynamic HTML, או DHTML,** היא שיטה ליצירת דפי [HTML](http://he.wikipedia.org/wiki/HTML) אינטראקטיביים על ידי שילוב של HTML - שהיא השפה שמשמשת לבניית ועיצוב דפים לרשת ה[אינטרנט](http://he.wikipedia.org/wiki/%D7%90%D7%99%D7%A0%D7%98%D7%A8%D7%A0%D7%98) כמעט מאז ראשיתה, [שפת תסריט](http://he.wikipedia.org/wiki/%D7%A9%D7%A4%D7%AA_%D7%AA%D7%A1%D7%A8%D7%99%D7%98) שעובדת בצד הלקוח (כמו [Javascript](http://he.wikipedia.org/wiki/Javascript)), שימוש ב־[DOM](http://he.wikipedia.org/wiki/DOM) המאפשר התייחסות לכל תג HTML כאל אובייקט ופעמים רבות גם שימוש ב־[Cascading Style Sheets](http://he.wikipedia.org/wiki/Cascading_Style_Sheets) על מנת להגדיר מאפיינים בתצוגה של העמוד.
* בדרך זו ניתן להציג בדף האינטרנט תפריטים הנפתחים ונסגרים באופן דינאמי, תאים המשנים את צבעם בעת מעבר סמן העכבר מעליהם וכדומה.
* את הפעולות מבצעת תוכנת ה[דפדפן](http://he.wikipedia.org/wiki/%D7%93%D7%A4%D7%93%D7%A4%D7%9F) על גבי מחשב הגולש, עובדה שפירושה לרוב זמן תגובה קצר יותר (המידע לא צריך להישלח ממחשב הגולש אל ה[שרת](http://he.wikipedia.org/wiki/%D7%A9%D7%A8%D7%AA) וחזרה, וכן נחסכת טעינת המסמך מחדש במחשב הגולש).
* הבעייתיות בשימוש בדפי DHTML היא העובדה שעל אף קיומו של תקן שנקבע על ידי ה־[W3C](http://he.wikipedia.org/wiki/W3C) דפדפנים שונים מתייחסים בצורה שונה למאפיינים של ה־[DOM](http://he.wikipedia.org/wiki/DOM) ולפקודות [JavaScript](http://he.wikipedia.org/wiki/JavaScript) ומוסיפים מאפיינים ופקודות שאינן קיימות בתקן על מנת ליצור אפשרויות חדשות. שונות זו מקשה על המעצב ליצור דף DHTML שיוצג כראוי בכל הדפדפנים.

DOM

* **Document Object Model**, 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

**AJAX**

* במסגרת אותו דף אני יכול לקבל נתונים נוספים.
* Call back function שם גנרי שאני קובע מה יקרה שאני מפעיל אותה ומה יקרה לפונקציה שאנחנו כותבים אותה והיא קוראת לעצמה ומשתמשת בה כאשר קורה משהו והיא **מופעלת על ידי מערכת.**
* אפשר לשלוח שאילתא בעזרת OPEN AND SEND
* רק האובייקט של אג'אקס שולח בקשה והוא מקבלה תגובה חזרה.
* לצורך הדוגמא בחיפוש של גוגל אנחנו כותבים מילה והוא משלים לנו אותה אז בעצם זאת הפעולה שאג'אקס עושה הוא שולח שאילתה על המידע הנוסף ואז חוזרת התגובה ומציגה את האפשרות המלאה.
* אני מקבל נתונים סופיים מהפונקציה ואנחנו יכולים לדעת אם הנתונים תקינים.
* **הפונקציה מופעלת כל הזמן. עד 4 פעמים בכל שאילתא...ו2 מינימום...**
* האם אני בצורה מפורשת מפעיל פונקציות תגובה לקבלת נתונים משרת של מנוע אג'אקס? לא! המערכת מפעילה.
* מה התפקיד של אג'אקס? האם חובה להשתמש במערכת באג'אקס?
* למה הוא נקרא אנסיכרוני? כי אני שולח בלי לקבל תשובה.
* חסרונות- לא עובד כפתור החזרה ואי אפשר לחזור כי אז נהרס כל הנתונים ששמרנו.  
  יתרון – שיש עדכון אוטומטי של נתונים בלי לרענן דף.
* **האם זה כל מה שיש ב-AJAX - לשלוח בקשת HTTP, עדכון דף "חי"? כן. עם זאת, זו רק אבן הבניין הבסיסית ביותר. נוסף על כך, יש מסגרות, שיטות עבודה מומלצות, תבניות עיצוב ועוד ...**
* העיקרון המנחה של **AJAX** אלטרנטיבה, יישומים מועברים כדפי אינטרנט.
* AJAX אינה טכנולוגיה בפני עצמה. זוהי טכניקה פיתוח אינטרנט, שילוב טכנולוגיות קיימות מספר באופן ספציפי: HTML, JavaScript, ובמיוחד XMLHttpRequest, ו (אופציונלי) XML.
* יישומי AJAX מספקים רמה גבוהה יותר של אינטראקטיביות עם משתמשי הקצה. שלא כמו תוכן דינמי, בתבנית HTML יכול המשתמש לקיים אינטראקציה עם תוכן בצד השרת - בלי לטעון מחדש את העמוד כולו.
* RIA - יישומי אינטרנט עשירים. Web 2.0 - דינמי / אינטראקטיבי יישומי אינטרנט. שניהם מבוססים בעיקר על AJAX.
* **AJAX, יתרונות -** קשובים יותר למשתמשים . צמצום העומס על השרת. שימוש בחוכמה למשל, רק את החלקים החדשים במפה נטענים ב-Google Maps, רק תשובות חסרות בטופס ייטענו שוב לקלט, "פורום", וכו '
* מדוע להשתמש ב-AJAX? בעיקר כדי לבנות אתר אינטרנט מהיר, דינמי, אך גם לחסוך משאבים. לשיפור שיתוף המשאבים, עדיף להשתמש בכוח של כל מחשבי הלקוח, ולא רק בשרת רשת ייחודי. AJAX
* מאפשרת לבצע עיבוד במחשב הלקוח (ב-JavaScript) עם נתונים שנלקחו מהשרת.
* העיבוד של דפי האינטרנט נעשה בעבר רק בצד השרת, תוך שימוש בשירותי אינטרנט או סקריפטים בצד השרת, לפני שכל הדף נשלח אל הרשת. AJAX יכול לשנות באופן סלקטיבי חלק של הדף מוצג על ידי הדפדפן, ולעדכן אותו ללא צורך לטעון מחדש את המסמך כולו עם כל התמונות, תפריטים וכו '.. לדוגמה, בתחומי טפסים, אפשרויות של המשתמשים, וכו ', יכול להיות מעובד והתוצאה מוצגת מיד באותו דף.
* מגביר את השימושיות של יישומי אינטרנט, מה שהופך אותם כביכול למהירי תגובה יעיל יותר אינטראקטיבי, ובכך משפר את חוויית המשתמש. חוסך רוחב פס - הורדות הנתונים המבוקשים רק לאחרונה, ולא את הדף כולו. ממשקים מהירים יותר - אם משתמשים בו נכון ולא מוגזם על עמוד אחד.
* **AJAX, חסרונות** – אין תמיכה בלחצן חזרה (במידה מסוימת, את הבעיה ניתן לפתור).
* **אם JavaScript אינו מופעל, AJAX לא יכול לעבוד**.
* על המשתמש מתבקש להגדיר JavaScript מבין האפשרויות של הדפדפן, מאחר שניתן להציג נתונים נטען באופן דינמי, זה לא חלק של הדף, ואת מילות המפתח בפנים לא נמצאים בשימוש על ידי מנועי החיפוש.
* כתובת האתר לא משתנה כאשר המצב של היישום משתנה.
* במצב אסינכרוני הדף עשוי להשתנות עם עיכובים (כאשר העיבוד בשרת לוקח כמה פעמים), זה עשוי להיות מטריד.
* בעיות בדפדפנים עלולות להופיע,
* השימוש הנרחב JavaScript בדף עלול להאט מעבדים ישנים.
* עלול להיות מבוטל או לא נראה בטעמי ביטחון או מסוג הדפדפן.
* **אם הבקשה נקרא XMLHttpRequest, התוכן חייב להיות XML?** לא נכון .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
* 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

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.

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.

http\_request.onreadystatechange = function() {

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

return; // *Just ignore earlier states – common practice.*

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

alret(“Server reported error code: ” +

http\_request.status);

return;

}

**n = document.getElementById(“nodeID”);**

**n.innerHTML = http\_request.responseText;**

};

http\_request.open(“GET”, “http://x.com/foo”, true);

http\_request.send(null);

* מחלקה XMLHttpRequest תכונות:

readyState: קוד ברצף משנה את הערך 0-4 עבור כל מצב האובייקט XMLHttpRequest. סטטוס: 200 בסדר, 404 אם הדף לא נמצא.

responseText: מחזיק נתונים שנטענו כמחרוזת של תווים.

responseXml: מחזיק קובץ שנטען XML, בשיטה של DOM המאפשרת לחלץ נתונים.

onreadystatechange: המאפיין לוקח את הערך ומופעל כאשר האירוע readystatechange נשלח.

* שיטות:

פתח - לאתחל את בקשה חדשה האובייקט.

שימוש: פתוח (מצב url, [בוליאני], [שם משתמש, סיסמה])

מצב: סוג של בקשה, קבל או פרסם.

כתובת האתר: את מיקום קובץ, באמצעות נתיב. בוליאני: נכון (אסינכרוני) / לא נכון (סינכרוני). שם משתמש וסיסמה.

* עוד שיטות:

בטל () - מבטל את הבקשה הנוכחית.

getAllResponseHeaders (headerName) - מחזיר את כותרות HTTP כמחרוזת. headerName: מחזירה את הערך של הכותרת העליונה מסוג HTTP שצוין. עוברים null או none, להחזיר את סט שלם של כותרות HTTP כמחרוזת. setRequestHeader (ערך המותג) - מוסיף זוג תווית / ערך לכותרת מסוג HTTP לשליחה.

* על ידי הגדרת המאפיין innerHTML של אלמנט, אלמנט יכול לשנות את תוכנו. התוכן הוא מה שבין <tag> ו </ tag>. בדרך כלל משתמשים עם תגי <div> ו <span>.תוכן חדש יכול לכלול HTML.
* **GetElementById** פונקציה ניתן להשתמש בהם כדי לשנות כל מרכיב בכלל :

<b **id="message"**>Hello!</b><br />

<span onclick="**getElementById('message').innerHTML='Bye!'"**>

Click here.

</span>

* אפשרויות כיסוי לכל סוגי הדפדפנים :

if (window.XMLHttpRequest) // Mozilla, Safari,...

http\_request = new XMLHttpRequest();

else if (window.ActiveXObject) { // IE

try {

http\_request = new ActiveXObject("Msxml2.XMLHTTP");

}

catch (e) {

try {

http\_request = new ActiveXObject("Microsoft.XMLHTTP");

} catch (e) {}

}

}

if (!http\_request) {

// … try to fail gracefully …

}

* כדי לשלוח את הבקשה, אנו משתמשים בשיטות פתוחות ולשלוח את הבקשה של אובייקט:

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

http\_request.send(null);

**JSON**

* גישה שנותנת להגדיר אובייקטים בגאווה סקריפט כמחרוזת.
* חשוב מאוד למערכות מידע מודרניות.
* משתמשים כתקן פופולארי ביותר להעברת נתונים באפליקציות שונות.
* -JSON נעשה שימוש נרחב להעברת מבני מידע ברשת בתהליך הקרוי [סריאליזציה](http://he.wikipedia.org/wiki/%D7%A1%D7%A8%D7%99%D7%90%D7%9C%D7%99%D7%96%D7%A6%D7%99%D7%94). השימוש העיקרי בפורמט הוא במסגרת תכנות [AJAX](http://he.wikipedia.org/wiki/AJAX_(%D7%AA%D7%9B%D7%A0%D7%95%D7%AA)), שם משמש JSON תחליף אפשרי לפורמט [XML](http://he.wikipedia.org/wiki/XML).
* 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.**
* מה קובע את התקן זה? אם יש לי סוגריים מסולסלים זה אובייקט.
* אם הם מרובעים זה מערך.
* מערך הוא סוג של אובייקט.
* **ייחודית בכך שהיא שפה שקלה מאוד לקריאה!**
* מינימלי טקסטואלי **תת קבוצה של JavaScript**
* JSON (סימון JavaScript Object) הוא פורמט לשינויי DATA קל משקל קל לבני אדם לקריאה ולכתיבה.
* 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.
* אנחנו יכולים להשתמש באופרטור חדש ליצירת אובייקטים של JavaScript
* אנחנו יכולים לגשת את המאפיינים ישירות, אנחנו יכולים להוסיף רכיבים באופן דינאמי לדף האינטרנט שלנו.
* האובייקטים לא מסודרים בקבוצות , אובייקטים עטופים {} "," מפריד מפתח / ערך זוגות ":" מפריד מפתחות וערכים המפתחות הם מחרוזות הערכים הם ערכים JSON
* **{**

**"name": "Jack B. Nimble",**

**"at large": true,**

**"grade": "A",**

**"format": {**

**"type": "rect",**

**"width": 1920,**

**"height": 1080,**

**"interlace": false,**

**"framerate": 24**

**}**

**}**

* 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.

**jQuery**

* 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.
* אין מקום רציני שלא משתמש בספרייה הזאת.
* הדולר זה השם של הפונקציה וגם השם של האובייקט.
* דולר מקבל סלקטור ושם אפשר לכתוב $(“a:first”) הוא פונה לאיבר הראשון שנקרא a.
* **האם דולר מחזיר אובייקט או אובייקט שעוטף מערך אלמנטים? הוא מחזיר אובייקט אלמנטים של HTML.שעוטף מערך**
* $(“a:first”) . באמצעות נקודה אחרי אפשר להשתמש בפונקציות של מובנות בג'יי קוורי
* **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
* כלי עזר - כמו גירסת הדפדפן ואת כל פונקציה
* מניפולציה CSS
* אפקטים ואנימציות
* Ajax
* אירועים
* הרחבה
* 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...

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 בתו הסגירה מתאים**
6. **בתג <a> אפשר להשתמש ליצירת קשר לחלקים שונים במסמך html**

2. מה נכון לגבי אלמנט div

1. **אלמנט div בונה פיסקה**
2. **אלמנט div אינו כולל עיצוב מיוחד כברירת מחדל**
3. **תוכן של אלמנטdiv אפשר לשנות בצורה דינאמית ע"י שינוי מאפיין INNERHTML**
4. לאלמנט divיש תפקיד שונה מתפקידו של אלמנט div
5. במסמך html יכול להיות אלמנט div אחד בלבד

3. מה נכון לגבי אלמנט FORM במסמך ? HTML

1. **באלמנט FORM אחד יכולים להיות מספר כפתורים מסוג SUMBIT**
2. **אלמנטים INPUT מיועדים להיות בתוך אלמנט FORM**
3. **לכל אלמנט INPUT נותנים שם לפיו מעבדים נתונים המתקבלים מטופס בשרת**
4. קליק על כפתור SUMBIT גורם לשליחת נתוני הטופס לשרת והעברה לדף HTML אחר
5. במסמך HTML אחד יכול להיות אלמנט FORM אחד בלבד

4. מה נכון בנוגע למשתנים ב JAVASCRIPT

1. **אם המשתנה מוגדר מחוץ לפונקציה תחום הכרתו כל מסמך כולל חלק שבתוך אלמנט BODY**
2. משתנים אפשר להגדיר רק בתוך חלק של הקוד שנמצא באלמנט HEAD של מסמך HTML
3. **למשתנה שמוגדר בתוך פונקציה ללא שימוש בהגדרה var יש גישה גם מחוץ לפונקציה**
4. למשתנים אין סוג נתונים
5. **למשתנה שקיבל ערך מספרי אפשר בהמשך לתת ערך של מחרוזת**

5. מה נכון בנוגע לתכנות מונחה עצמים ב-JAVASCRIPT

1. **לבניית אובייקטים משתמשים באופרטור new**
2. אי אפשר להגדיר בנאי
3. **להגדרת מחלקה לא משתמשים במילה השמורה class**
4. **שימוש ב json מאפשר דרך להגדרת קבוצת נתונים בצורת אובייקט**
5. **גישה לנתונים חברי מחלקה אפשר לקבל גם באמצעות אופרטור [ ]**

6. מה נכון בנוגע ל AJAX

1. מגדירים פונקציית תגובה שמנגנוןAJAX 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 חייבים להיות בצורת מסמך XML

7. מה נכון לגבי מערכים ב JAVASCRIPT

1. **משתמשים בסוגרים מרובעים [ ] לגישה לאלמנטים של מערך**
2. **להתחלת מערך משתמשים ב () new array**
3. **למערך יש תכונה בשם length**
4. במערכים אינדקס מתחיל מ-0 וחייב להשתמש עד למספר ברצף מקסימלי
5. כל הערכים במערך חייבים להיות באותו סוג

8. מה נכון לגבי שורת קוד הבאה document.getElementByld(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 getElementByld(a) אפשר לכתוב getelementbyld(a)
5. **a חייב להיות משתנה שערכו מוגדר לפני שורת קוד**

9. מה נכון לגבי HTML ו- DOM

1. **במאפיין id של אלמנט משתמשים לקבלת גישה לאלמנט כאובייקט ב-javascript**
2. 2 מילים של טקסט במסמך של HTML הן 2 אובייקטים בDOM
3. DHTML זה שפה נוספת ל-HTML
4. **לכל אלמנט של HTML אפשר לקבל גישה כאובייקט ב- javascript**
5. **כל אלמנט של HTML הוא אובייקט ב-DOM אפילו אם לא מוגדר לו id**

10. מה נכון בנוגע למשפט הבא בקובץ JSP ? <%=a.toString()%>

1. המשפט שגוי כי אין ; בסוף
2. אפשר להחליף את המשפט בביטוי זהה ב-javascript
3. המשפט יכול להיות ממוקם בקובץ רק אחרי תווית <body >
4. **משתנה a חייב להיות מוגדר היטב לפני המשפט**
5. **המשפט יוצר מחרוזת במסמך שנשלח למשתמש הקצה**

11. מה נכון לגבי השימוש של JSP ו- SERVLET

1. שרת התומך ב SERVLET חייב לתמוך ב JSP
2. אפשר לפתח יישום web שמשתמש ב JSP ללא תרגום קבצי JSP
3. **אפשר לפתוח יישום web שמשתמש ב SERVLET ללא שימוש ב JSP**
4. **שרת התומך ב JSP חייב לתמוך ב SERVLET**