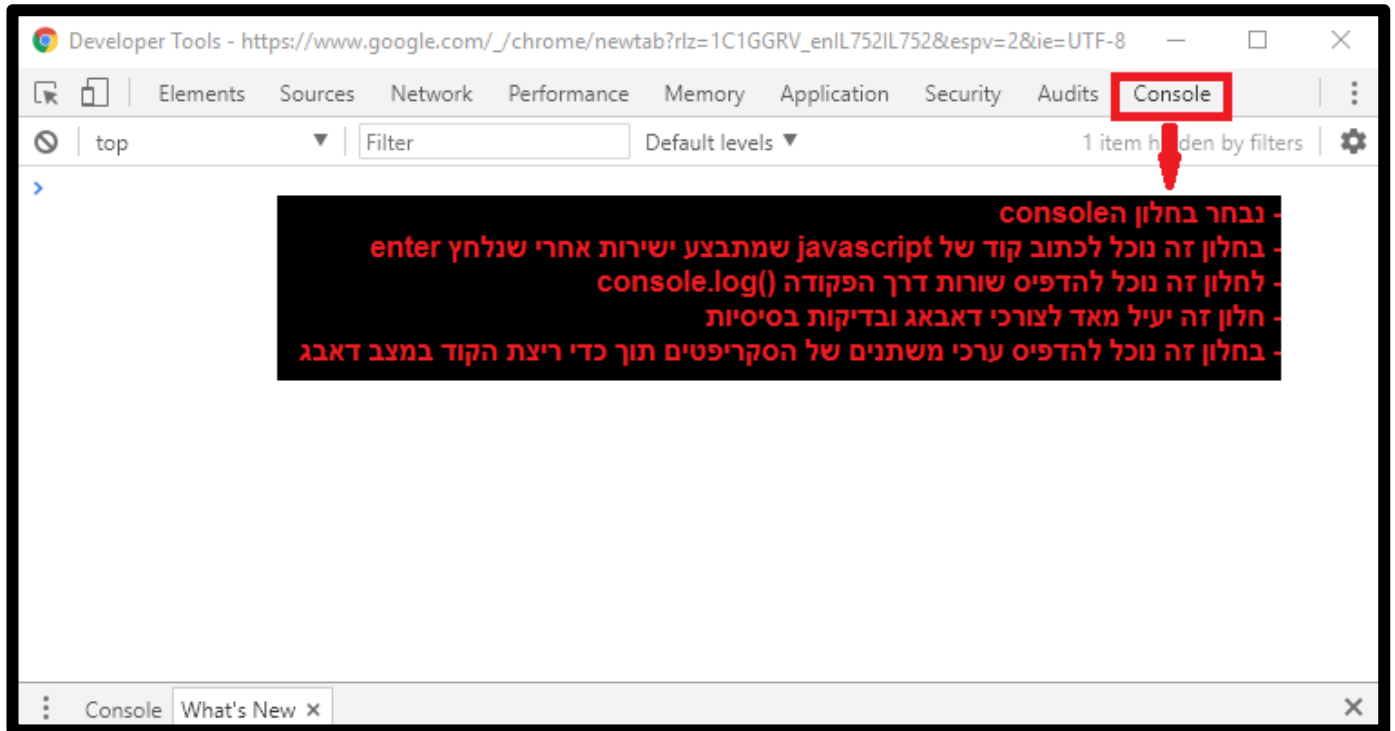


javascript

JavaScript מספקת לעמוד HTML יכולת להשתנות ולהגיב לפעולות המשתמש אחרי שהדף כבר נטען בדפדפן.
(לדוגמא: תמונות יתחלפו כשתעבור עליהן עם העכבר, שדות טפסים יכולים להשפיע זה על זה בהתאם למה שתמלא בהם וחישובים יכולים להתבצע)

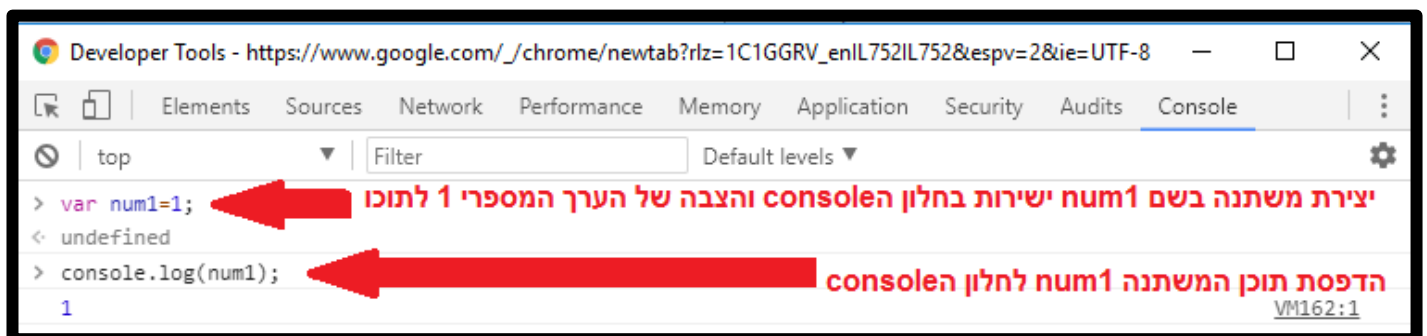
עבודה עם javascript ישירות מתוך חלון console שנפתח ע"י לחיצה על 12F

שלב 1:



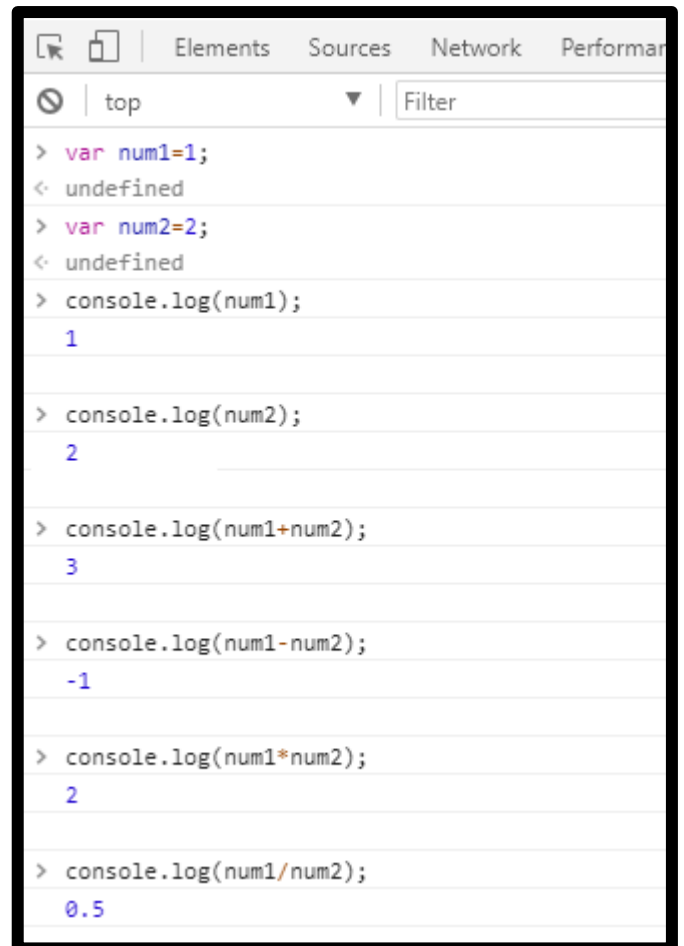
שלב 2:

יצרנו משתנה אחד פשוט, הצבנו לתוכו ערך, והדפסנו את תוכן המשתנה



שלב 3:

יצרנו שני משתנים והדפסנו את תוצאות חיבור / חיסור / כפל / חילוק ביניהם.



The screenshot shows a web browser's developer console with the 'Sources' tab selected. The console displays the following JavaScript code and its output:

```
> var num1=1;
< undefined
> var num2=2;
< undefined
> console.log(num1);
1
> console.log(num2);
2
> console.log(num1+num2);
3
> console.log(num1-num2);
-1
> console.log(num1*num2);
2
> console.log(num1/num2);
0.5
```

סוגי המשתנים בjavascript

בjavascript לא מגדירים למשתנה סוג של טיפוס מסויים, ולכן כל משתנה יכול להכיל את כל סוגי המשתנים, ואפשר להחליף לו את סוג התוכן (לדוגמה בהתחלה המשתנה יכיל מספר, ואחר כך נציב לתוכו מחרוזת).

נראה זאת בדוגמא הבאה שבה כתבנו את דף ה html הבא:

```

<!DOCTYPE html>

<html lang="en">
<head>
  <meta charset="utf-8" />
  <title></title>
  <script>

    // סוגי המשתנים האפשריים
    var a = 123;
    document.write("a = " + a + " type = " + typeof (a) + "<br/>");
    var a = "hello";
    document.write("a = " + a + " type = " + typeof (a) + "<br/>");
    var a = true;
    document.write("a = " + a + " type = " + typeof (a) + "<br/>");
    var a = new Date();
    document.write("a = " + a + " type = " + typeof (a) + "<br/>");
    var a = new Array(7,12,8,1);
    document.write("a = " + a + " type = " + typeof (a) + "<br/>");
    var a = null;
    document.write("a = " + a + " type = " + typeof (a) + "<br/>");

    function f() {
      alert("function message");
    }

    var a = f;
    document.write("a = " + a + " type = " + typeof (a) + "<br/>");

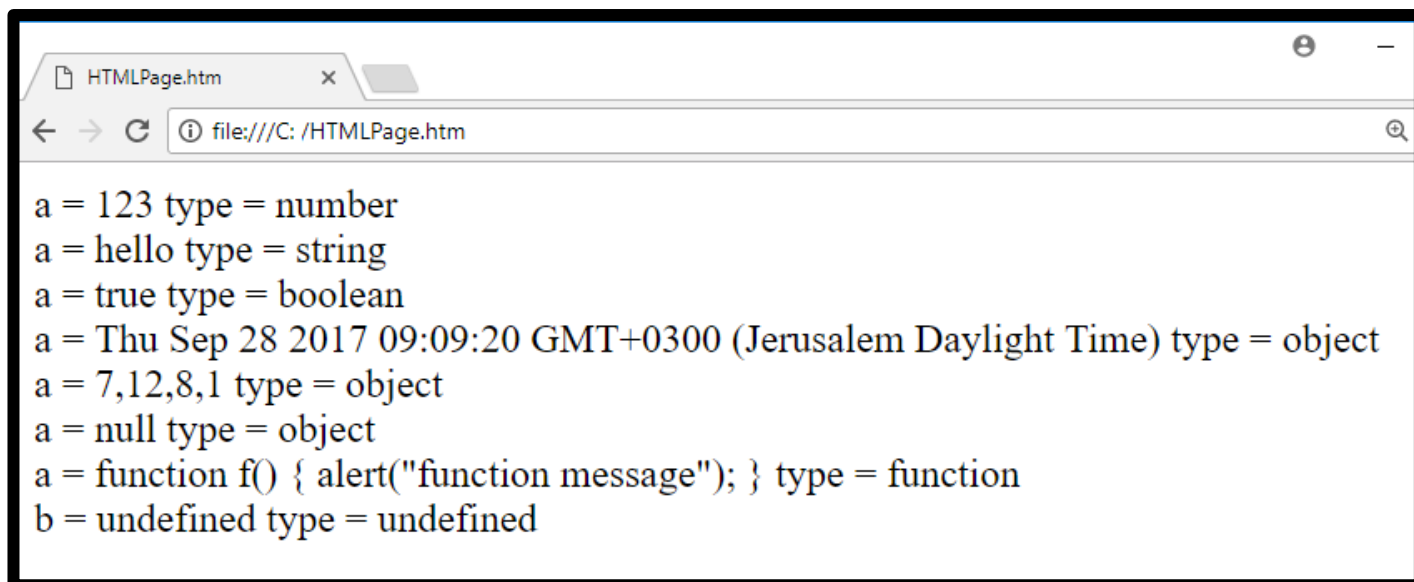
    var b;
    document.write("b = " + b + " type = " + typeof (b) + "<br/>");

  </script>
</head>
<body>

</body>
</html>

```

בהרצת הדף קיבלנו את הפלט הבא:



הערה חשובה:

כאשר יצרנו פונקציה, ניתן להציב את ה פונקציה לתוך משתנה (כאן הצבנו את f לתוך a). ואז ע"י המשתנה שמכיל את הפונקציה, לבצע קריאה לפונקציה:

```
<!DOCTYPE html>

<html lang="en">
<head>
  <meta charset="utf-8" />
  <title></title>
  <script>

    סוגי המשתנים האפשריים

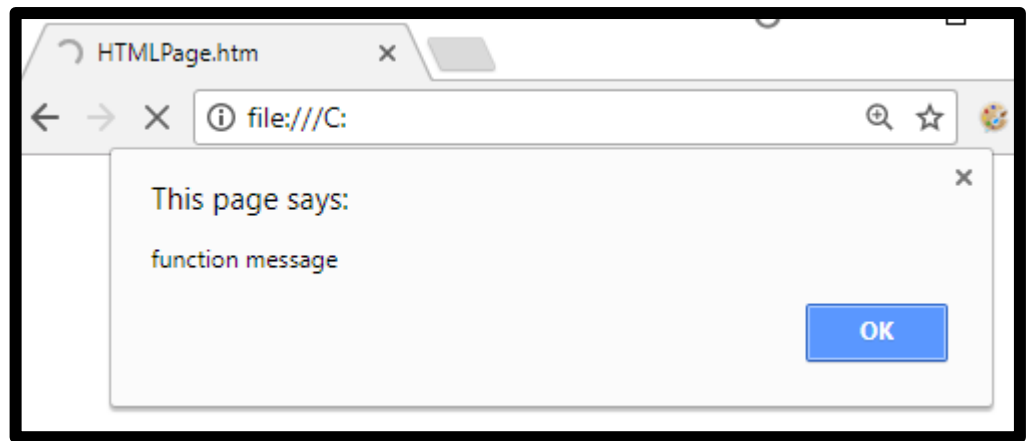
    function f() {
      alert("function message");
    }

    var a = f;
    a();

  </script>
</head>
<body>

</body>
</html>
```

פלט התכנית:



שלבי טעינת דף html

הערה חשובה: document.write שמתבצע אחרי טעינת הדף, דורס את כל התוכן של הדף (כלומר: כל התוכן שהיה קיים בדף ימחק, ורק מה שכתבנו ע"י ה document.write יופיע בדף)

בדוגמא הבאה יצרנו document.write שכותב ישירות מתוך תגית script, ולכן נראה שהביצוע של ה document.write יהיה עוד לפני שהדף נטען, ולכן תוכן הדף לא ידרס:

```
<!DOCTYPE html>

<html lang="en">
<head>
  <meta charset="utf-8" />
  <title></title>
  <script>

    document.write("before page");

  </script>
</head>
<body>
  <h1>HTML PAGE</h1>
  <button onclick="document.write('clicked')">click me</button>
</body>
</html>
```

תוצאת ההרצה של הדף



הסבר: מכיוון שהdocument.write התבצע עוד לפני טעינת הדף, הוא קודם כל נכתב לדף, ולא דרס את תגית הכותרת והכפתור, כי הם עוד לא נטענו לדף.

לאחר שהdocument.write הסתיים להתבצע, הדף המשיך להיטען, והגיע לחלק של ה body שמכיל את התגיות כותרת וכפתור, שהתווספו לדף שכבר היה כתוב בו "before page"

נתקדם, עוד צעד, ונראה איך אפשר לבצע קוד מיד בסיום הטעינה:

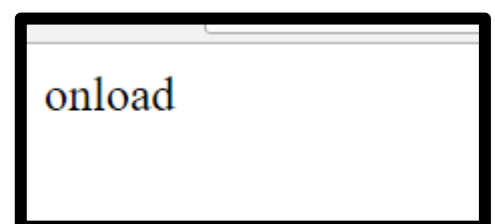
הביצוע יהיה באירוע onload שייכתב בתוך התגית של ה body ויתבצע מיד כשהbody סיים להיטען (בתוך האירוע נכלל לרשום או קריאה לפונקציה, או קוד javascript רגיל, כאן פשוט נרשום קוד javascript ישירות לתוך האירוע):

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="utf-8" />
  <title></title>
  <script>

    document.write("before page");

  </script>
</head>
<body onload="document.write('onload') ">
  <h1>HTML PAGE</h1>
  <button onclick="document.write('clicked') ">click me</button>
</body>
</html>
```

תוצאת הרצת הדף:



הסבר: מכיוון שהdocument.write התבצע בסיום טעינת הדף, הוא דרס את תגית הכותרת והכפתור, שכבר נטענו לדף.

נראה דוגמא דומה, בה השתמשנו באירוע onclick במקום באירוע onload, במקרה הזה, הדף יטען באופן רגיל, ונראה את הכותרת והכפתור, אבל כאשר נלחץ על הכפתור ויתבצע האירוע onclick שמבצע document.write ידרוס את תוכן כל הדף:

```
<!DOCTYPE html>

<html lang="en">
<head>
  <meta charset="utf-8" />
  <title></title>
  <script>

    document.write("before page");

  </script>
</head>
<body>
<h1>HTML PAGE</h1>
<button onclick="document.write('clicked')">click me</button>
</body>
</html>
```

תוצאת ההרצה של הדף



אחרי לחיצה על הכפתור הדף יכיל:



נבצע תרגיל נוסף, כדי להבין שהbody בכלל עוד לא נטען לפני האירוע onload

נשתמש באובייקט document.body – אובייקט javascript שמובנה אוטומטית בכל דף שרץ בדפדפן ומכיל את תוכן body של הדף הספציפי שרץ.

נדפיס את תוכן document.body לחלון ה console בעזרת הפקודה console.log.

את ההדפסה הזו נבצע בשלושה מקומות שונים:

1. לפני האירוע onload- נכתוב את ההדפסה ישירות בתוך התגית של script וההדפסה תתבצע לפני שהדף נטען
2. באירוע onload – ההדפסה תתבצע בסיום טעינת הדף
3. באירוע onclick – ההדפסה הזו תתבצע בכל פעם שנלחץ על הכפתור

להלן עמוד הhtml שיצרנו:

```
<!DOCTYPE html>

<html lang="en">
<head>
  <meta charset="utf-8" />
  <title></title>
  <script>

    console.log (document.body) ;

  </script>
</head>
<body onload="console.log (document.body) ">
  <h1>HTML PAGE</h1>
  <button onclick="console.log (document.body) ">click me</button>
</body>
</html>
```

כאשר הרצנו את הדף, קיבלנו את התוצאה הבאה:



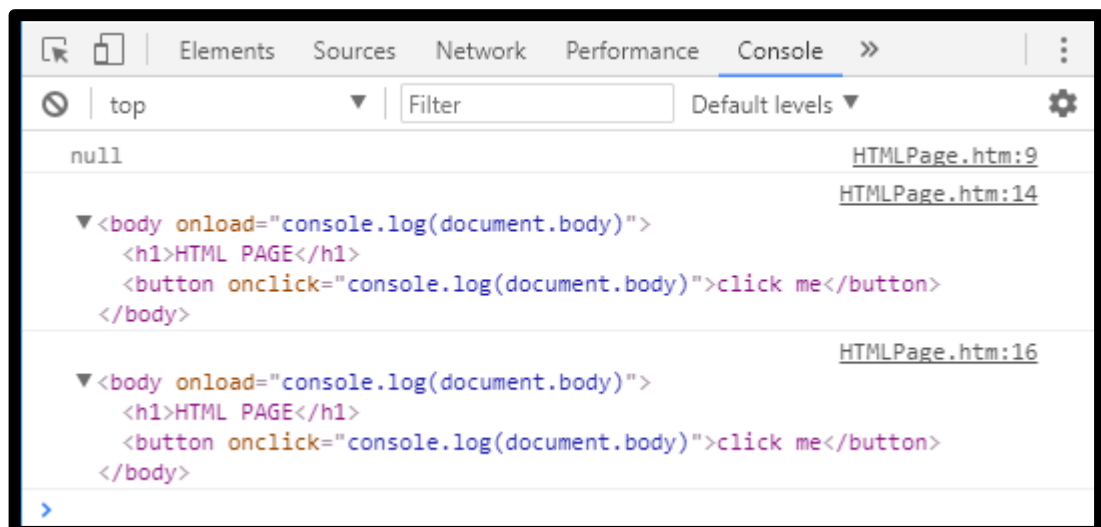
ותוכן חלון ה Console יהיה:



שימו לב שיש רק שני הדפסות, כי ההדפסה לפני טעינת העמוד, וההדפסה בטעינת העמוד התרחשו באופן אוטומטי ללא צורך בפעולה של המשתמש שמתבונן בדף.

אבל ההדפסה שמתרחשת בלחיצה על הכפתור עוד לא התרחשה כי המשתמש עדיין לא לחץ על הכפתור.

נלחץ על הכפתור, וכעת הconsole יכיל את התוכן הבא:



נוכל לראות שתוכן הdocument.body באירוע onclick ובאירוע onload זהה לחלוטין, ומכיל את תוכן הbody בדיוק כפי שציפינו.

לעומת זאת – לפני האירוע onload הbody מכיל null – כי הוא עוד לא נטען.

פונקציות בתגיות script

כאשר ניצור פונקציות בתוך script, הם לא יתבצעו אוטומטית כמו הפקודה console.log שרשמנו מקודם ישירות בתגית script. הסיבה היא שפונקציות לא מתרחשות עד שנקרא להם. ולכן כאשר נריץ את עמוד הhtml הבא:

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="utf-8" />
  <title></title>

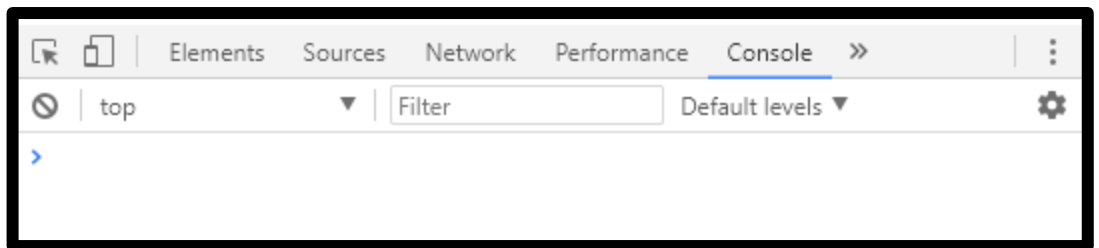
  <script>
    function f1() {
      console.log("onload");
    }
    function f2() {
      console.log("onclick");
    }
  </script>
</head>
<body>
  <h1>HTML PAGE</h1>
  <button>click me</button>
</body>
</html>
```

נקבל את התוצאה:

HTML PAGE

click me

אבל חלון הconsole יהיה ריק, ולא נרשם שם שום דבר ע"י הפונקציות:



לעומת זאת, כאשר נקרא לפונקציות מתוך האירועים:

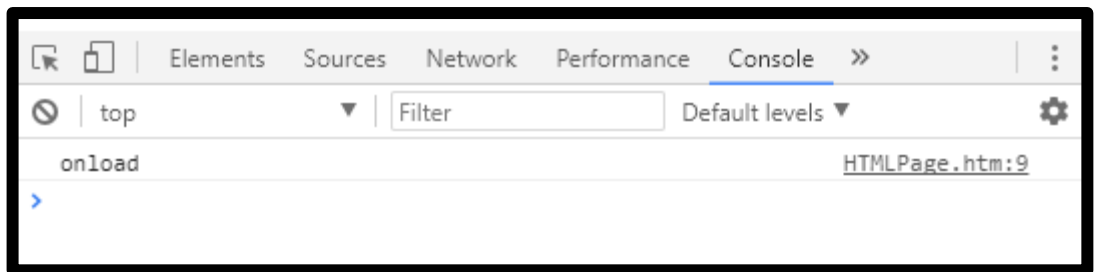
```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="utf-8" />
  <title></title>

  <script>
    function f1() {
      console.log("onload");
    }
    function f2() {
      console.log("onclick");
    }
  </script>
</head>
<body onload="f1()">
  <h1>HTML PAGE</h1>
  <button onclick="f2()">click me</button>
</body>
</html>
```

נקבל את התוצאה:



וחלון console יכיל את התוכן של הפונקציה שהתבצעה אוטומטית בסיום טעינת הדף באירוע onload:



נלחץ כעת על הכפתור, שיקרא לפונקציה f2 שתוסיף גם היא פלט לחלון console:

