תכנות באינטרנט

Java Script

גלעד מרקמן

קריית החינוך פארק המדע, נס ציונה

Microsoft

ASP.NET Core

GM

Internet Programing Courses

קריית החינוך השש שנתית פארק המדע עתיד של חדשנות

מבוא לתכנות צד לקוח
JavaScript

JavaScript

- JavaScript היא שפת תכנות המיועדת לדפי אינטרנט, ונועדה לאפשר יצירת דפים דינמיים המגיבים לקלט מהמשתמש.

- שפת תסריט (Script) היא שפה המורצת באופן מיידי מהקוד, ואין צורך לבצע הידור (קומפילציה). שפת JavaScript מורצת על ידי הדפדפן ללא ביצוע של הידור קודם לכן.

- שפת JavaScript ושפת Java אילו הן שתי שפות שונות לחלוטין ואין להתבלבל ביניהם.

מיקום קוד JavaScript

- נתן לכתוב את קוד JS בשני אופנים:
  - בתוך תגית <script> - Internal
  - בקובץ נפרד וקישור הקובץ לדף External – HTML - הדרך המומלצת.

- תגית <script> יכולה להיכתב הן בראש העמוד (בתוך תגית <head>) והן בגוף העמוד (בתוך תגית <body>).

- כיוון שאנחנו כותבים את הקוד בתוך דפי Razor, אם רוצים להכניס את הקוד לכותרת <head> יש להשתמש בפקודה @section בתבנית.

תוכנית ראשונה

```
@page
@model RazorApp.Pages.JavaScript.HelloWorldModel
@{
}
<script type="text/javascript">
    alert("Hello World!")
</script>
```

- כמקובל נתחיל בתוכנית Hello World
- הקוד נכתב בתגית <script>
  - הוספנו את המאפיין type. אין חובה להוסיף אותו.
- הפקודה alert יוצרת חלון התראה עם ההודעה הכתובה בסוגריים.
- כל פקודה ב JS מסתיימת בנקודה פסיק (;).
- הקוד מורץ מייד כשהדף עולה !!!!

section - מיקוד קוד

-

```
@page
@model RazorApp.Pages.JavaScript.HelloWorldModel
@{
}
<script type="text/javascript">
    alert("Hello World!")
</script>

@section Scripts {
    <script type="text/javascript">
        alert("Hello World! - Section")
    </script>
}
```

```
<!DOCTYPE html>
<html lang="en">
<head>…</head>
<body>
    <header>…</header>
    <div class="container">
        <main role="main" class="pb-3">
            @RenderBody()
        </main>
    </div>

    <footer class="border-top footer text-muted">…</footer>

    <script src="~/lib/jquery/dist/jquery.min.js"></script>
    <script src="~/lib/bootstrap/dist/js/bootstrap.bundle.min.js"></script>
    <script src="~/js/site.js" asp-append-version="true"></script>

    @await RenderSectionAsync("Scripts", required: false)

</body>
</html>
```

כתיבת קוד JS בקובץ נפרד

- כתיבת קוד בקובץ נפרד (External) נעשית לפי השלבים הבאים:
  - יצירת ספריה מיוחדת לקבצי קוד
  - יצירת קובץ עם סיומת js. בתוך הספריה.
  - כתיבת הקוד בקובץ החדש <u>ללא</u> צורך בתגית <script>.
  - קישור קובץ ה JS לקובץ HTML באמצעות הפקודה:

```
<script src="../JS Code/JavaScript.js"></script>
```

- ניתן לקשר מספר קבצי קוד JS לדף אינטרנט.
- צירוף הקוד בגוף הדף או באמצעות section בכל מקום שנבחר.

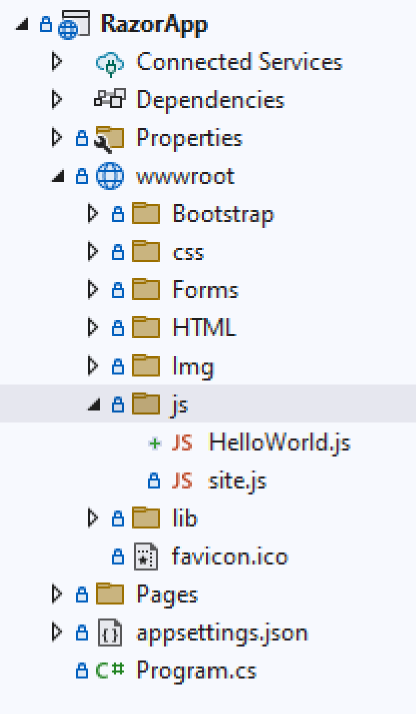
כתיבת קוד JS בקובץ חיצוני

- קוד JS נכתב בספריה wwwroot/js
  - בקובץ HelloWorld.js.
- החיבור נעשה באמצעות פקודת <script>

HelloWorld.js

RazorApp JavaScript Content Files

```javascript
alert("Hello World!");
```



HelloWorld_External.cshtml

RazorApp

```cshtml
@page
@model RazorApp.Pages.JavaScript.HelloWorld_ExternalModel
@{
}

<script src="/js/HelloWorld.js"> </script>
```

פקודות בסיסיות

משתנים

- בגאווה סקריפט ניתן להצהיר על משתנה בדרכים הבאות:

רמות המשתנים (scope):

משתנים ב JS הם גלובאליים (למשל המשתנה X)

var - משתנה ברמת פונקציה/פעולה.

let - משתנה ברמת ה block.

```
x = 5;
var name = "Gilad";
let num = 3.15;

var v1;
let v2
```

סוגי משתנים

- המשתנים בגאווה סקריפט הם דינמיים. אין צורך להכריז מהו סוג המשתנה והסוג נקבע לפי הערך המוצב בתוך המשתנה:

```
var int = 5;                     // typeof = number
var real = 5.13;                 // typeof = number
var str = "my Name is Gilad";    // typeof = string
var bool = true;                 // typeof = boolean
var v1;                          // typeof = undefined
```

- אם נציב למשתנה ערך מסוג אחר – המשתנה יהפוך דינמית לסוג החדש. אין צורך בהכרזות מקדימות.
- משתנה שרק הוכרז ולא הוצב בו ערך הוא מסוג undefined.

כתיבה לדף

- ניתן גם לכתוב לתוך דף HTML באמצעות הפקודה הבאה:

```
<script>
    document.writeln("Hello world");

    document.write("<h1 style='text-align: center'> Hello world </h1>");
</script>
```

- שימו לב – ניתן לכתוב לדף גם תגיות HTML ולא רק טקסט.
- אין הבדל משמעותי בין write ל- writeln שכן כפי שהסברנו בעבר HTML מתעלם ממעברי שורה.
- כתיבה למקום מסויים בדף – נלמד בהמשך.

קליטת נתונים מהמשתמש () prompt

- ישנם מספר דרכים לקלוט נתונים מהמשתמש. אחד הדרכים היא באמצעות הפקודה (”...“)prompt. הפקודה מחזירה ערך string.

```
var a, b, c;
a = prompt("Enter first number");
b = prompt("Enter second number");
c = a + b;
document.write(c);
```

- כדי לקלוט מספרים עלינו להוסיף פקודה ההופכת string למספר ()parseInt.

```
var a, b;
a = parseInt(prompt("Enter first number"));
b = parseInt(prompt("Enter second number"));
document.write("a + b =" + (a + b));
```

פעולות חשבוניות על משתנים

- ניתן לבצע באמצעות המשתנים פעולות חשבוניות, לדוגמה:
- שימו לב ש = זהו סימן להצבה ולא לשוויון.
- ניתן להציב באותו משתנה מספר ערכים שונים. בכל הצבה נמחק הערך הקודם.

```
var num1 = 5;
var num2 = 7;
var res = num1 + num2;
document.writeln(res);
document.write("<br>");
res = num1 * num2 / 2;
document.write(res);
document.write("<br>");
res++;
document.write(res);
document.write("<br>");
```

חיבור מחרוזת

- פעולת החיבור + בין משתנים שהם מחרוזות מבצעת "שרשור" של המחרוזת.
- חיבור בין מספר למחרוזת הופכת את התוצאה למחרוזת.
- רישום מספר בסוגריים הופך אותו למחרוזת.

```
var str1 = "Gilad";
var str2 = ",";
var str3 = " Yoav";
var str = str1 + str2 + str3; // "Gilad, Yoav"
str = str3, str1, str2; // " YoavGilad,"
```

```
var str1 = "Gilad";
var num = 1;
var res = str1 + num; // "Gilad1"
```

```
var num1 = "1";
var num2 = "2";
var res = num1 + num2; // "12"
```

פעולות הצבה וצוברים

- ראינו כי האופרטור = מבצע הצבה בתוך משתנה. לכן אין מניעה לכתוב את הפקודה הבאות:

```
var a = 5;
var b = 2;
a = a + 1; // a == 6
a = a * a; // a == 36
a = a - b; // a == 34
```

- קיצורי הדרך הבאים הם מאוד נפוצים:

```
a++;  // a = a + 1;
a--;  // a= a - 1;
```

תנאים

- תנאי if-else ותנאי switch-case:

```
switch(expression) {
  case x:
    // code block
    break;
  case y:
    // code block
    break;
  default:
    // code block
}
```

```
if (condition1) {
    // code to execute if condition1 is true;
} else if (condition2) {
    // code to execute if condition1 is false and condition2 is true;
} else {
     // code to execute if condition1 and condition2 are false;
}
```

אופרטורים השוואתיים ובוליאנים

- התנאים (condition) יכולים לכלול את האופרטורים הבאים:

JavaScript Comparison Operators

Operator	Description
==	equal to
===	equal value and equal type
!=	not equal
!==	not equal value or not equal type
>	greater than
<	less than
>=	greater than or equal to
<=	less than or equal to
?	ternary operator

JavaScript Logical Operators

Operator	Description
&&	logical and
||	logical or
!	logical not

לולאות

- בגאווה סקריפט קיימות כל הלולאות המקובלות בשפות תכנות, נביא כדוגמה שתיים מהן:
- לולאת for

```
alert("start");
for (var i = 1; i < 10; i++) {
    document.write(i);
}
```

- לולאת while

```
alert("start");
var i = 1;
while (i < 10) {
    document.write(i);
    i++;
}
```

פונקציות (פעולה)

- פונקציה פשוטה (ללא פרמטרים וללא החזרת ערך):

```
function funcName() {
    // Code to excute
}
```

- פונקציה עם פרמטרים:

```
function funcName(p1, p2) {
    // Code to excute
}
```

- פונקציה עם פרמטרים המחזירה ערך :

```
function funcName(p1, p2) {
    // Code to excute
    return result
}
```

כתיבת הערות ותיעוד

- הערה comment כותבים בשתי דרכים:

```
/*
    multi line comment
 */

alert("hello world"); // comment
```

- ניתן לכתוב או לבטל הערה גם באמצעות הכפתור המיוחד של Visual Studio.

- ניתן לעטוף קוד בהערה על מנת למנוע את הרצתו לצורך בדיקה debugging.

```
/*alert("hello world");*/
```

אוביקטים ומחלקות

אובייקטים ללא מחלקות

- הגדרת אובייקט ללא צורך במחלקה.

```
var person = {
    firstName: "Gilad",
    lastName: "Markman",
    year: 1968,
    city: "Rehovot",
    student: true,
    Age: function () {
        return 2020 - this.year;
    }
};
```

```
alert(person.name);
person.name = prompt("Enter New Name");
```

בנאי של אובייקט

```
function Person(first, last) {

    this.firstName = first;
    this.lastName = last;
    this.year = 1900;
    this.city = "";
    this.student = null;
    this.Age = function () {
        return 2021 - this.year;
    }
}

var student1 = new Person("Noa", "Levy");
var student2 = new Person();
student2.firstName = "Gali";
student2.lastName = "Ron";
```

- יצירת אובייקט חדש על בסיס תבנית של אובייקט.
- הפונקציה היא הבנאי של האובייקט.

מחלקה class

```
alert("start");
class Person {
    constructor(first, last) {
        this.firstName = first;         // public
        this.lastName = last;
        this.year = 1900;
        this.city = "";
        this.student = null;
    }

    Age () {
        return 2021 - this.year;
    }
}

var student = new Person("Noa", "Levy");
alert(student.firstName);
```

- החל מ- 2015 ניתן להגדיר מחלקות.
- לא מדובר בתכנות מונחה עצמים.
- אין כימוס ואין הורשה.

גאווה סקריפט ו- HTML
אירועים

פונקציות ואירועים

- קוד JS מורץ מייד כשהדף עולה. אנחנו רוצים שהקוד יופעל (יזומן) רק בהתקיים אירוע מסויים: המשתמש לחץ על כפתור, המשתמש שלח טופס לשרת וכד'.
- לצורך כך אנחנו נעזרים בפונקציות ואירועים:
  - הפונקציה מופעלת רק כאשר מזמנים אותה.
  - הזימון נעשה בעקבות אירוע בדף, ובדרך זו אנחנו יוצרים דף HTML דינמי.
- דוגמאות לאירועים:
  - לחיצה על העכבר על תגית מסוימת - onclick
  - לחיצה כפולה על תגית - ondblclick.
  - סיום העלאת הדף - onload.
  - שינוי בשדה input – onchange
  - לחיצה על המקלדת בתיבת טקסט – onkeydown
  - ועוד ...

קריאה לפונקציה בהתקיים אירוע

- על מנת להשתמש באירוע אנחנו מבצעים את הפעולות הבאות:
  - מגדירים את הארוע בדף HTML. הארוע שייך לתגית מסויימת בדף.
  - כותבים פונקציה שתופעל בעת שהתרחש האירוע
- דוגמה, לחיצה על כפתור:

```
<body>
    <form id="form1">
        <input type="button" id="btn" name="btn" value="click" onclick="clickFunc()"/>
    </form>
</body>
```

- הפונקציה שתופעל בעת הלחיצה:

```
function clickFunc() {
    alert("onclick");
}
```

דוגמאות לאירועים

- זימון פונקציה – באמצעות אירוע (event).

Event	Description
onchange	An HTML element has been changed
onclick	The user clicks an HTML element
onmouseover	The user moves the mouse over an HTML element
onmouseout	The user moves the mouse away from an HTML element
onkeydown	The user pushes a keyboard key
onload	The browser has finished loading the page

The list is much longer: W3Schools JavaScript Reference HTML DOM Events.

- https://www.w3schools.com/jsref/dom_obj_event.asp

onchange

- שימוש באירוע onchange של תיבת טקסט.
- דף HTML:

```
<!-- text -->
<label for="phone">phone:</label>
<input type="text" id="phone" name="phone" onchange="phoneValidate()" />
```

- הפונקציה:

```
function phoneValidate() {
    alert("event was fired");
}
```

Input events

- onblur – האלמנט מאבד את הפוקוס
- onfocus – האלמנט מקבל את הפוקוס
- onchange – נעשה שינוי באלמנט
- oninput – המשתמש הכניס נתונים לאלמנט
- onkeydown – המשתמש לחץ על מקש במקלדת
- onkeyup – המשתמש שחרר את המקש במקלדת
- onkeypress – המשתמש לחץ על מקש

פרמטרים של אירוע

- בעת קריאה לפונקציה לאחר אירוע (event) ניתן להעביר את האירוע כפרמטר לפונקציה לקבלת מידע נוסף. לדוגמה קבלת המקש שנלחץ ב onkeypress:

```html
<!--event parameters-->
<label for="phone">event parameters example</label>
<input type="text" id="text" name="text" onkeypress="keyPressFun(event)" />
```

```javascript
function keyPressFun(event) {
    alert("key: " + event.key + ",  keyCode:" + event.keyCode);
    if (event.key <= "0" || event.key >= "9") {
        alert("not digit");
        //return false;
    } else
        alert("digit");
    //return true;
}
```

ביטול אירוע

- ניתן לבטל את האירוע באמצעות JS ולהחזיר את המצב לפני האירוע, באמצעות החזרת false מהפונקציה.
- לדוגמא, ביטול הקשת התו על ידי המשתמש אם אינו מספר:

```
<input type="text" id="text" name="text" onkeypress="return keyPressFun(event)" />
```

```
function keyPressFun(event) {
    alert("key: " + event.key + ",  keyCode:" + event.keyCode);
    if (event.key <= "0" || event.key >= "9") {
        alert("not digit");
        return false;
    } else
        alert("digit");
    return true;
}
```

- ביטול אירוע onclick יבטל את משלוח הדף לשרת !!!!!!

אינטראקציה עם דף HTML

אינטראציה עם HTML

- באמצעות JavaScript אנחנו יכולים ליצור אינטראקציה עם דף HTML. לקרוא מה כתוב בדף ולשנות את האוביקטים בדף.

- ציינו שלושה סוגים של משתנים ב JavaScript:
  - מספר number
  - מחרוזת string
  - בולאני boolean.

- משתנה בגאווה סקריפט יכול להיות גם מסוג אובייקט של HTML. כלומר, משתנה יכול להיות מצביע ל"תגית" או אלמנט של HTML שכתבנו בדף.

Document.getElmentById ()

- על מנת לקרוא או לשנות אובייקט (תגית) בדף HTML אנחנו משתמשים בפקודה הבאה:

```
var x = document.getElementById("tagId");
```

- הפקודה מוצאת את האלמנט בדף עם ה- id המבוקש, ומציבה אותו בתוך משתנה x.

- לאחר מכן אנחנו יכולים לקרוא את המאפיינים של האלמנט או לשנות אותם.

innerHTML

- כתיבה למקום מסויים בסף נעשה באמצעות עדכון המאפיין innerHTML:

```html
<body>
    <button type="button" id="btn" onclick="btnChange()">change Div</button>
    <br /> <br />
    <div style = "background-color:lightblue; width:300px; height:200px" id="myDiv">

    </div>
</body>
```

- קוד JavaScript:

```javascript
function btnChange() {
    var div = document.getElementById("myDiv");
    div.innerHTML = "<p> New content </p>"
}
```

- בקיצור ניתן לכתוב:

```javascript
document.getElementById("myDiv").innerHTML = "<p> New content </p>";
```

שינוי עיצוב

- באמצעות JS ניתן לשנות גם עיצוב CSS כך:

```
function changeStyle() {
    var div = document.getElementById("myDiv");
    div.style.color = "red";
    div.style.backgroundColor = "blue";
    div.style.border = "solid 3px black";
    div.style.verticalAlign = "center";
}
```

- https://www.w3schools.com/jsref/dom_obj_style.asp

toggle color – דוגמה נוספת

• שינוי צבע של אוביקט בהתאם למצבו.

```
function toggleColor() {
    var div = document.getElementById("myDiv");
    if (div.style.color == "red") {
        div.style.color = "blue";
        div.style.backgroundColor = "red";
    }
    else {
        div.style.color = "red";
        div.style.backgroundColor = "blue";
    }
```

דוגמה – תיבת Select ושינוי צבעים

- דוגמה לקריאה של תיבת select.
- שינוי הצבע בתיבה ישנה צבע בדף.

```html
<!-- select -->
<label for="colors">Choose a color:</label>
<select id="colors" name="colors" onchange="selectFun()">
    <option value="null" selected="selected"></option>
    <option value="red">Red</option>
    <option value="blue">Blue</option>
    <option value="green">Green</option>
    <option value="yellow">Yellow</option>
</select>
<br /> <br />
<div id="div2"></div>
```

```javascript
function selectFun() {
    var select = document.getElementById("colors");
    var div = document.getElementById("div2");
    var myDiv = document.getElementById("myDiv");

    var i = select.selectedIndex;
    var color = select.options[i].value;
    div.innerHTML = "select index= " + i;
    div.innerHTML = div.innerHTML + "  item = " + color;
    myDiv.style.backgroundColor = color;
}
```

מחרוזות – בדיקת טופס
Regular Expression

קריאת תו במחרוזת

- ניתן לפנות לתו במחרוזת בשתי דרכים:
  - באמצעות הפונקציה ()charAt.
  - באמצעות האינדקס וסוגריים מרובעים [0]str.

```
var ch1 = str[i];
var ch2 = str.charAt(i);
```

- <u>רשימת פעולות על מחרוזות:</u>
- https://www.w3schools.com/js/js_string_methods.asp
- https://www.w3schools.com/jsref/jsref_obj_string.asp

פונקציות מחרוזת נבחרות

החזרת אורך המחרוזת	s1.length();	
החזרת תו במקום k במחרוזת	s1.charAt(k);	תו ראשון נמצא במקום 0
החזרת המיקום הראשון של תו/תחילת תת-מחרוזת מתחילת המחרוזת. (-1) אם לא קיים.	s1.indexOf(ch) s1.indexOf(str)	ch – תו str – מחרוזת מיקום ראשון = 0
החזרת המיקום הראשון של תו/תחילת תת-מחרוזת ממיקום k במחרוזת (-1) אם לא קיים.	s1.indexOf(ch, k) s1.indexOf(str, k)	ch – תו str – מחרוזת מיקום ראשון = 0
החזרת המיקום האחרון של תו/תחילת תת-מחרוזת מסוף המחרוזת. (-1) אם לא קיים.	s1.lastIndexOf(ch) s1.lastIndexOf(str)	ch – תו str – מחרוזת מיקום ראשון = 0
החזרת המיקום האחרון של תו/תחילת תת-מחרוזת ממיקום k מסוף המחרוזת במחרוזת. (-1) אם לא קיים.	s1.lastIndexOf(ch, k) s1.lastIndexOf(str, k)	ch – תו str – מחרוזת מיקום ראשון = 0
מחליף את כל אותיות המחרוזת לאותיות קטנות	s1 = s1.toLowerCase();	
מחליף את כל אותיות המחרוזת לאותיות גדולות	s1 = s1.toUpperCase();	

Regular Expression

- Regular Expression – היא שיטה לבדיקה אם תבנית מסויימת של תווים מתקיימת במחרוזת נתונה.
- הדרך לשימוש ב RegExp היא כך:
  - מגדירים אוביקט מסוג RegExp הכולל תבנית מסויימת של תווים.
  - באמצעות פעולות ()test או ()match אנו מבצעים בדיקה אם התבנית שהוגדרה מתקיימת במחרוזת הנתונה.
- קיימים כללים לבניית התבנית באמצעות תווים מוסכמים. לפירוט הכללים ראו:
- https://www.w3resource.com/javascript/form/email-validation.php
- https://www.w3schools.com/jsref/jsref_obj_regexp.asp

Regular Expression

- לצורך הדגמת השיטה נבנה את הטופס הבא:

```html
<form id="form1">
    <label for="text">text:</label>
    <input type="text" name="txt" id="txt"/>
    <br /> <br />
    <div id="msg" style="width:400px; height:100px; border:solid"></div>
    <br /> <br />
    <input type="button" value="check" name="btn" id="btn" onclick="chkText()"/>
</form>
```

- לחיצה על כפתור check תריץ את הפונקציה.

Regular Expression

- בדיקה אם במחרוזת קיים רצף של התווים "abc" במקום כלשהו.
- תבנית regExp מוגדרת באמצעות /.../.

```
function chkText() {
    var text = document.getElementById("txt").value;
    var msg = document.getElementById("msg");
    var reg = /abc/;
    if (reg.test(text))
        msg.innerHTML = "True";
    else
        msg.innerHTML = "False";
}
```

- בדיקה אם המחרוזת מתחילה ב abc

```
var reg = /^abc/;
```

- בדיקה אם המחרוזת מסתיימת ב abc

```
var reg = /abc$/;
```

Regular Expression

- סוגריים מרובעים – האם המחרוזת מכילה לפחות אחד מהתווים.
- האם המחרוזת מכילה ספרות.
- המחרוזת מכילה אותיות גדולות או קטנות.
  - i - case-insensitive matching
- המחרוזת מכילה אותיות או ספרות.
- המחרוזת מכילה Gilad או Markman

```
var reg = /[2a8]/;
```

```
reg = /\d/; reg = /[0-9]/
```

```
reg = /[a-z]/i; reg = /[a-zA-Z]/
```

```
reg = /\w/; reg = /[a-zA-Z0-9]/;
```

```
reg = /(Gilad|Markman)/;
```

Regular Expression

```
reg = /^0\d{9}$/;
```

```
reg = /^[a-zA-Z]+@[a-zA-Z]{3,4}$/;
```

```
reg = /\W/;
```

```
reg = /\s/;
```

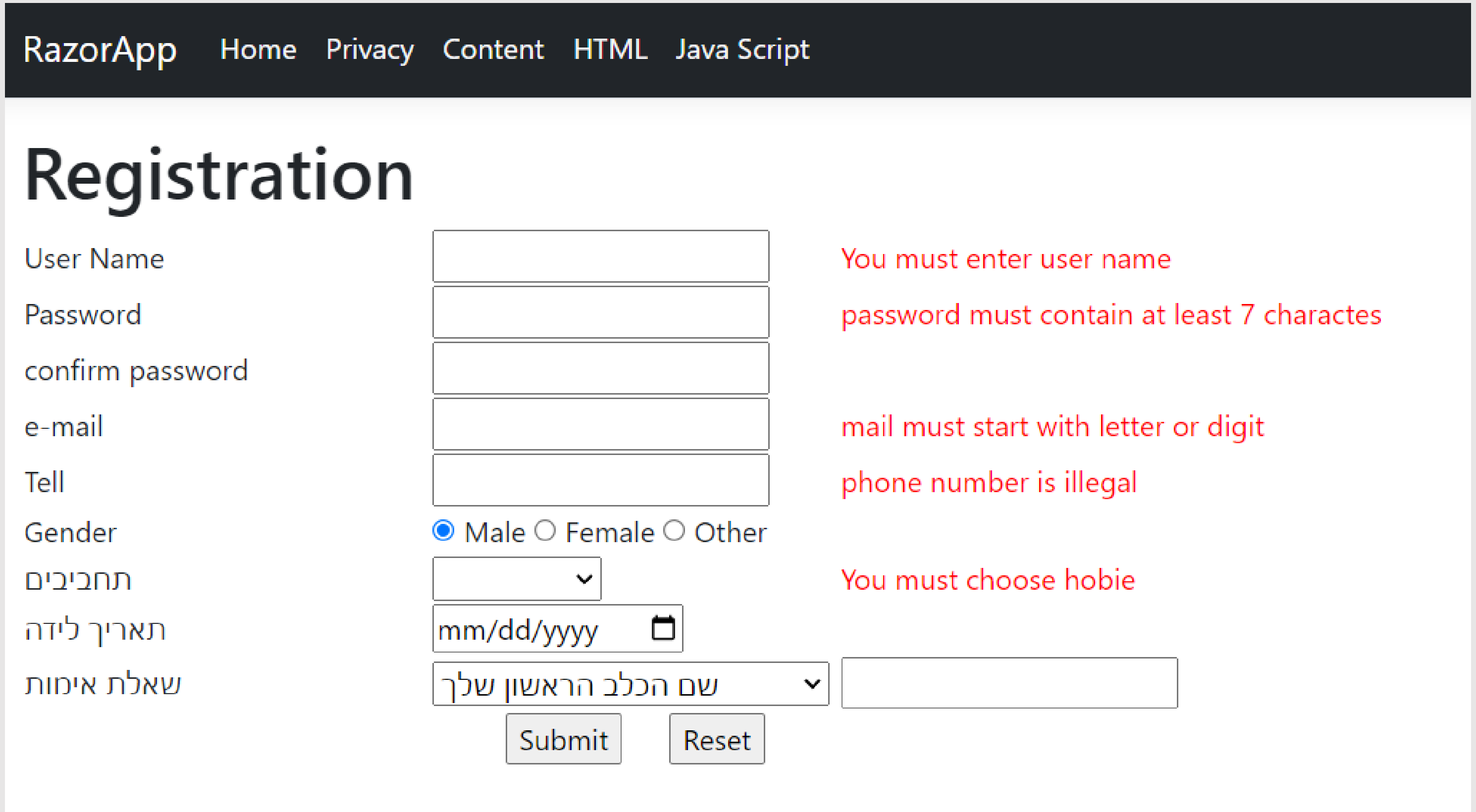
```
reg = /\D/;
```

- מחרוזת מתחילה בספרה 0 וכוללת לאחריה 9 ספרות.
- המחרוזת מתחילה באות אנגלית אחת או יותר (+), לאחר מכן מכילה תו @ אחד, ולאחריו 3-4 אותיות באנגלית.
- המחרוזת כוללת תווים מיוחדים (נכריח להוסיף תו מיוחד בסיסמה).
- המחרוזת כוללת רווח (למנוע רווחים במחרוזת).
- המחרוזת כוללת תווים שאינם ספרות (לבדוק שהוכנסו רק ספרות)

דוגמה
אימות טופס

אימות טופס

• בניית טופס באמצעות טבלה עם מקום להערות.



בניית פונקציה ראשית onsubmit

- הוספת אירוע onsubmit לטופס:

```
<form id="form1" onsubmit="return validateForm()">
    <table>
```

- הוספת אירוע onclick לכפתור submit:

```
<input type="submit" name="submit" onclick="return validateForm()" />
```

- בניית הפונקציה הראשית לבדיקת הטופס:
  - ביטול האירוע במקרה והפונקציה מחזירה false.
  - צבירה של תנאים בוליאנים באמצעות "וגם".
  - סדר האופרנדים (הפונקציה מבוצעת קודם ל-"וגם").

```
function validateForm() {

    var res = true;
    res = userNameVal() && res;
    res = passwordVal() && res;
    res = confirmFun() && res;
    res = confirmemail() && res;
    res = confirmPhone() && res;
    return res;
}
```

בדיקת User Name

```
function userNameVal() {
    var userName = document.getElementById("userName").value;
    var msgBox = document.getElementById("userNameMsg");
    if (userName.length == 0) {
        msgBox.innerHTML = "You must enter user name";
        return false;
    }
    if (!isLetter(userName[0])) {
        msgBox.innerHTML = "User name must start with a letter ";
        return false;
    }
    msgBox.innerHTML = "";
    return true;
}
```

- בדיקת אם הערך ריק.
- בדיקת אות התחלתית.
- כתיבת הערות ומחיקתם

בדיקת סיסמה

```javascript
function passwordVal() {
    var pass = document.getElementById("password").value;
    var msgBox = document.getElementById("passwordMsg");
    if (pass.length < 7) {
        msgBox.innerHTML = "password must contain at least 7 charactes";
        return false;
    }
    var specialChar = /[@!#$%^&*()-+]/; //regular Expretion
    if (!specialChar.test(pass)) {
        msgBox.innerHTML = "password must contain one special char";
        return false;
    }
    msgBox.innerHTML = "";
    return true;
}
```

- מספר תווים מינימלי
- קיום תו מיוחד.

בדיקה תוך כדי הזנה בתיבת טקסט

- הבדיקה תוך כדי הזנה (אירוע onchange) וגם בפונקציה הראשית.

```
<input type="password" name="confirm" id="confirm" onchange="confirmFun()" />
```

- הפונקציה לבדיקה:

```
function confirmFun() {
    var pass = document.getElementById("password").value;
    var confirm = document.getElementById("confirm").value;
    var msgBox = document.getElementById("confirmMsg");

    if (pass != confirm) {
        msgBox.innerHTML = "confirm password doesn't mutch password";
        return false;
    }
    msgBox.innerHTML = "";
    return true;
}
```

אימות טלפון

• בדיקה בעת אירוע oninput וכן בפונקציה הראשית:

```
<input type="text" name="phone" id="phone" oninput="confirmPhone()"/>
```

• הפונקציה- שימוש ב regExp:

```
function confirmPhone() {
    var phone = document.getElementById("phone").value;
    var msgBox = document.getElementById("phoneMsg");

    var reg = /^0{1}(2|3|4|6|8|9|5[2-8]|73)-[1-9]\d{6}$/;
    if (!reg.test(phone)) {
        msgBox.innerHTML = "phone number is illegal";
        return false;
    }
    msgBox.innerHTML = "";
    return true;
}
```

אימות מייל

```
function confirmemail() {

    var email = document.getElementById("email").value;
    var msgBox = document.getElementById("emailMsg");

    reg = /^\w+([\.-]?\w+)*@\w+([\.-]?\w+)*(\.\w{2,4})+$/;
    if (!reg.test(email)) {
        msgBox.innerHTML = "Invalid email";
        return false;
    }

    msgBox.innerHTML = "";
    return true;
}
```

- שימוש ב Regular Expression.
  RegExp.
- * 0 או יותר פעמים
- \+ 1 פעמים או יותר
- \w אות או ספרה
- ? 0 או 1 פעמים
- \. התו נקודה "."
- {2,4} 2 – 4 פעמים.

Debugging
DevTools – F12

DevTools – F12

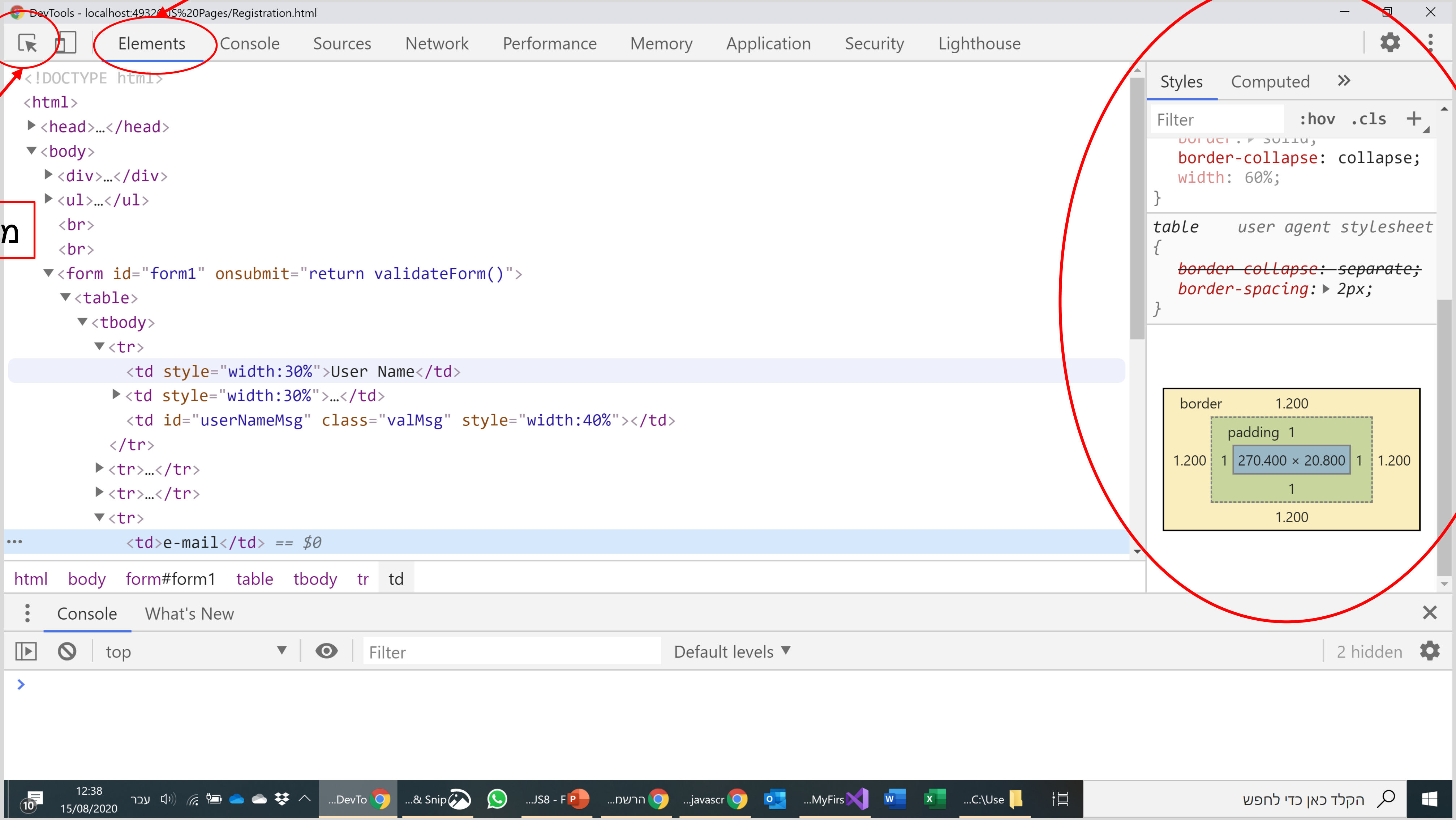
- מציאת טעויות ("באגים") בקוד היא פעולה סיזיפית ומתישה. לצורך כך קיימים בדפדנים כלי רב עוצמה העוזר לנו לבצע תיקון טעויות (debugging).
- אנחנו נדגים את הכלי של ה chrom, אולם בדפדפנים האחרים הכלים דומים.
- הכניסה לכלי הפיתוח נעשה מתוך הכרום על ידי לחיצה על F12.

מסך הפיתוח עיצוב

קוד HTML

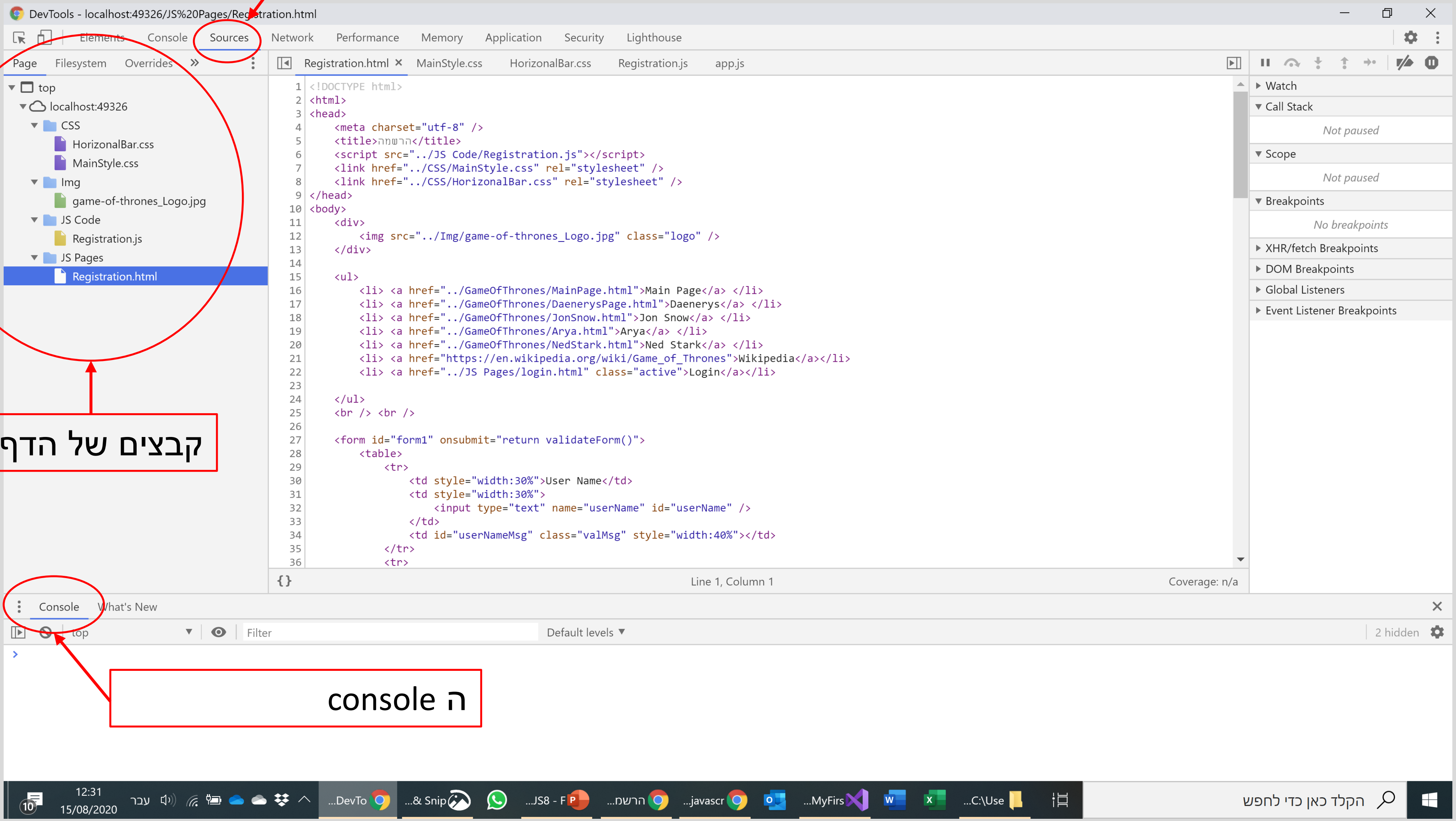
עיצוב CSS

מצביע לדף



מסך הפיתוח קוד

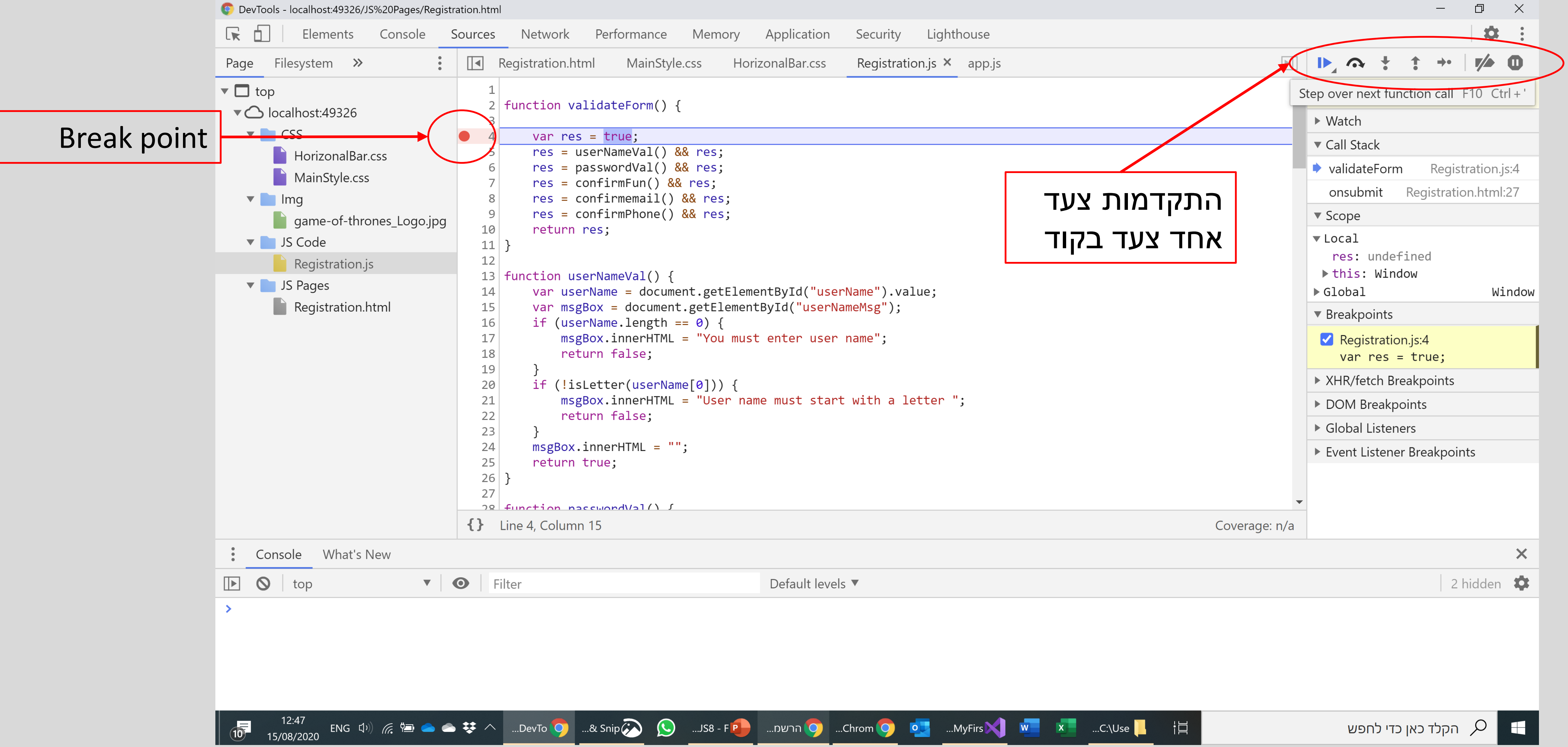
מעבר לקוד



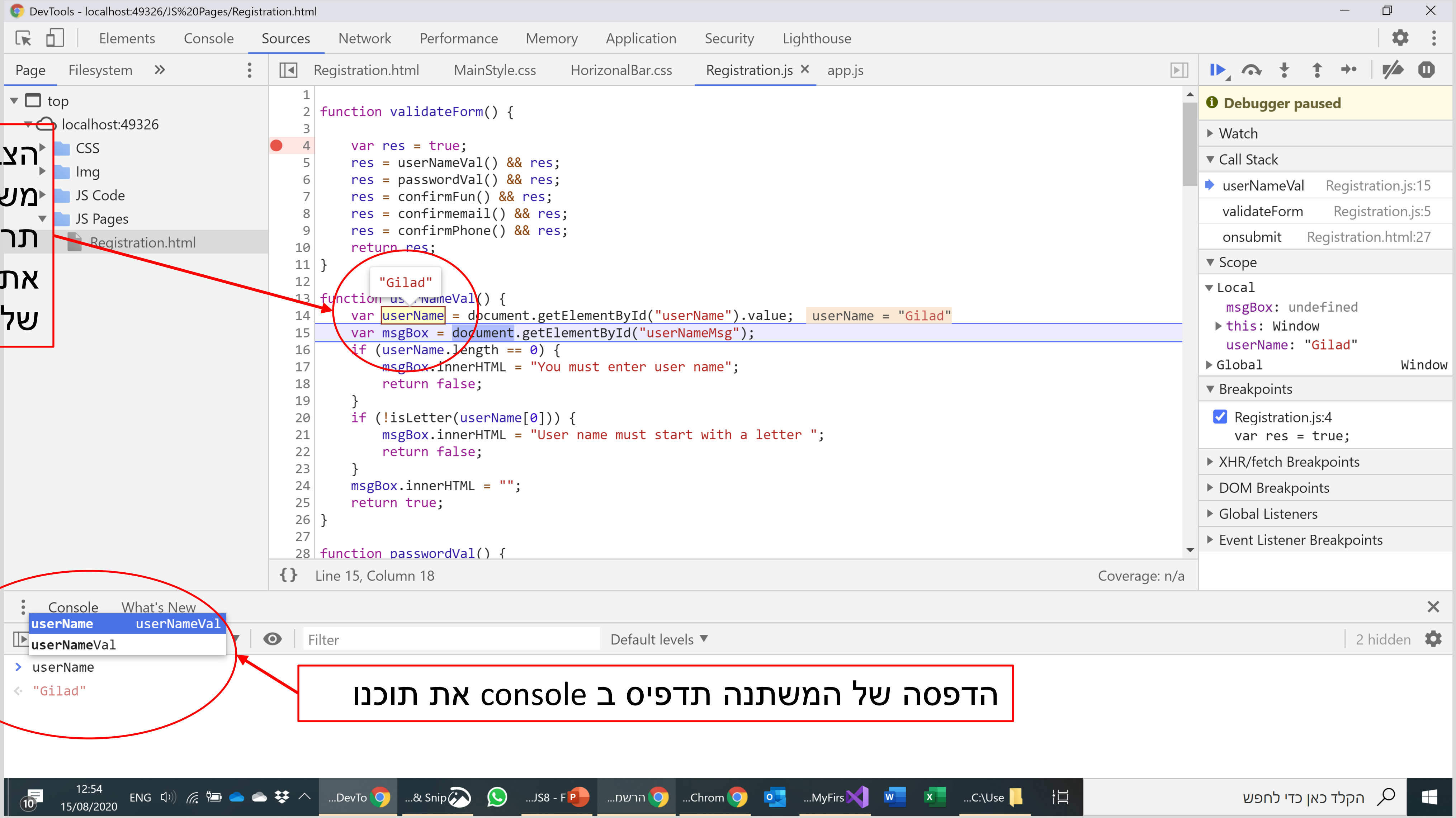
קבצים של הדף

console ה

javaScript Debugging



בדיקת תוכן משתנים



הצבעה על משתנה תראה לנו את התוכן שלו

הדפסה של המשתנה תדפיס ב console את תוכנו

סיכום

- הקדמה – מיקום כתיבת הקוד.
- פקודות בסיסיות ופונקציות.
- אוביקטים ומחלקות.
- אירועים Events.
- איטראקציה עם דף HTML – document.getElementById (“id”).
- מחרוזות RegExp
- אימות טפסים.
- DevTools - F12

GM
Internet Programing Courses

קריית החינוך
השש שנתית
פארק המדע

GM
Internet Programing Courses

קריית החינוך
השש שנתית
פארק המדע
עתיד של חדשנות

GM
Internet Programing Courses

קריית החינוך
השש שנתית
פארק המדע
עתיד של חדשנות

GM
Internet Programing Courses

קריית החינוך
השש שנתית
פארק המדע
עתיד של חדשנות

GM
Internet Programing Courses

קריית החינוך
השש שנתית
פארק המדע
עתיד של חדשנות

GM
Internet Programing Courses

קריית החינוך
השש שנתית
פארק המדע
עתיד של חדשנות

GM
Internet Programing
Courses

קריית החינוך
השש שנתית
פארק המדע
עתיד של חדשנות

GM
Internet Programing Courses

קריית החינוך
השש שנתית
פארק המדע
עתיד של חדשנות

GM
Internet Programing Courses

קריית החינוך
השש שנתית
פארק המדע
עתיד של חדשנות