



ראשי פרקים:

מבוא 

גרירת אלמנטים על המסך 

שמירת מידע ב Dom storage 

Notification 

קבצים 

WebWorker 

Modernizr 

Geolocation 

מבוא

HTML5 כוללת שורה של חידושים ב javascript. חידושים אלו מאפשרים יכולות שעד היום לא ניתן היה לממש אותם ב web, או בהשקעה גדולה מאוד של כתיבת קוד.

בין החידושים: יכולת גרירת אלמנטים על המסך באתר, גילוי מיקומו של הגולש לפי קוי אורך ורוחב של כדור הארץ ועוד.

גרירת אלמנטים על המסך

ב- HTML5 אפשרות הגרירה מוטמעת בתכונות ובאלמנטים השונים.

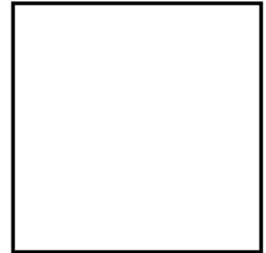
דוגמא:

בדוגמא זו גוררים את התמונה למסגרת.

לפני הגרירה:



גרור את התמונה למסגרת:



לאחר הגרירה:



גרור את התמונה למסגרת:



משימה

פתחי את התיקיה DragAndDrop והציגי בדפדפן את הדף: TrialDrag.htm
נסי לגרור את התמונה למסגרת.

כיצד ניתן לאפשר גרירת אלמנטים?

כדי לאפשר גרירה יש לכתוב פונקציות js עבור שלושת האירועים הבאים:

1. **ondragstart** – תחילת הגרירה.
2. **ondragover** – במהלך הגרירה.
3. **drop** – שחרור הגרירה.

ישנם אירועים נוספים הקשורים לגרירה, אך אין חובה לעורר אותם.

דוגמת קוד:

הדוגמא הבאה מציגה את דוגמת הגרירה שעיינת בה קודם.

Html:

על הדף מוצגים אלמנט ו- div (המסגרת). יש לגרור את התמונה ל div.

```
<body dir="rtl">
<p>למסגרת התמונה את גרור</p>
<div id="div1" ondrop="drop(event)" ondragover="allowDrop(event)" ></div>
<br /><br />

</body>
```

התמונה היא האובייקט הנגרר, ואילו ה div הוא האובייקט שאליו גוררים.

עבור התמונה יתרחש האירוע **ondragstart** המציין שהגרירה החלה. במהלך הגרירה המשתמש גורר את התמונה ל div, בעוד העכבר לחוץ, ואז יתעורר האירוע **ondragover**. סיום הגרירה הוא בשחרור העכבר והנחת התמונה ב div. אז יתעורר האירוע **ondrop**.

draggable התכונה

עבור האלמנט הנגרר (התמונה בדוגמא זו) יש להגדיר את התכונה **draggable** המאפשרת לאובייקט להיגרר. ללא תכונה זו, לא ניתן לגרור את האובייקט.

שימי לב: ניתן לגרור כל אלמנט: תמונה, פסקה, לחצן ועוד. אך יש לגרור אותם לאלמנט שיכול להכיל אלמנטים נוספים. לדוגמא: div. לא ניתן לגרור תיבת טקסט לתוך לחצן...

Java script:

בקוד js כותבים את הפונקציות שגורמות לתמונה הנגררת להכנס ל div.

```
<script type="text/javascript">
function drag(ev) {
    ev.dataTransfer.setData("Text", ev.target.id);
    ev.dataTransfer.effectAllowed = "move";
    ev.srcElement.style.top = ev.y + "px";
    ev.srcElement.style.left = ev.x + "px";
}

function allowDrop(ev) {
    ev.preventDefault();
}
```

```

}

function drop(ev) {
    ev.preventDefault();
    var data = ev.dataTransfer.getData("Text");
    ev.target.appendChild(document.getElementById(data));
}
</script>

```

הסבר הדוגמא:

הפונקציה drag

פונקציה זו מתרחשת בעת האירוע `ondragstart` – כאשר מתחילים לגרור את התמונה. הפונקציה מקבלת כפרמטר את האובייקט הנגרר. בדוגמא זו `ev` הוא התמונה הנגררת. (שימי לב בקריאה לפונקציה, בחלק של ה- `html`, יש לשלוח `event`, ראי דוגמא).

function drag(ev)

הפקודה הבאה מגדירה את האלמנט הנגרר:

ev.dataTransfer.setData("Text", ev.target.id);

הסבר:

האובייקט `dataTransfer` הוא האובייקט המטפל בגרירה.

לאובייקט זה שתי פונקציות: `setData`, `getData`.

הפונקציה `setData` נקראת כאן, הפונקציה `getData` תקרא בהמשך באירוע `drop`.

`setData(typeDrag, idDragElement)`

הפונקציה מקבלת את הפרמטרים הבאים:

1. `typeDrag` – אופן הגרירה. אנו תמיד נגדיר פרמטר זה כ- `"Text"`.

2. `idDragElement` – ערך ה- `id` של האלמנט הנגרר. ההגדרה היא **כללית** ולא נכתוב `"drag1"` (קוד

התמונה בדוגמא שלנו), אלא: `ev.target.id` – כך מקבלים את הקוד של האלמנט הנגרר.

קראי שוב את השורה ובדקי האם היא ברורה לך:

ev.dataTransfer.setData("Text", ev.target.id);

ניתן להגדיר את מראה העכבר בשעת הגרירה (מראה סטנדרטי של גרירה) ע"י הפקודה הבאה:

ev.dataTransfer.effectAllowed = "move";

שתי הפקודות הבאות אינן חובה. הן גורמות לאלמנט הנגרר להיצמד לעכבר בשעת הגרירה, לתוצאה יפה יותר.

ev.srcElement.style.top = ev.y + "px";

ev.srcElement.style.left = ev.x + "px";

הפונקציה allowDrop

פונקציה זו מתרחשת בעת האירוע `ondragover` - במהלך גרירת האלמנט מעל ה `div`

function allowDrop(ev)

הפקודה הבאה גורמת לכך שהדפדפן לא יטפל בנתונים הנגררים באופן שהוא רגיל, כברירת מחדל, אלא בצורה המתאימה.

ev.preventDefault();

הפונקציה drop

function drop(ev)

פונקציה זו מתרחשת בעת האירוע `ondrop` – כאשר משחררים את העכבר ומסיימים את הגרירה. הפקודה הבאה מבצעת כנ"ל

ev.preventDefault();

הפונקציה `getData` מחזירה את האובייקט שנשמר בפונקציה `setData` שהוסברה לעיל.

את האובייקט המוחזר שומרים בתוך משתנה שנקרא `data`.

לאחר פקודה זו `data` מכיל את התמונה שנגררה.

var data = ev.dataTransfer.getData("Text");

הפקודה הבאה שייכת ל `js` (ואינה קשורה ל `html5`) והיא בסה"כ מוסיפה את התמונה ל `div`.

ev.target.appendChild(document.getElementById(data));

ב- HTML5 נוספה אפשרות של הוספת תכונות מותאמות אישית. לכל אלמנט ישנן תכונות קבועות אשר משפיעות על התנהגות האלמנט. לדוגמא:

לאלמנט `img` התכונה `src` אשר קובעת מהי התמונה המוצגת. לאלמנט `input` התכונה `type` אשר קובעת מהו סוג הפקד. לעיתים יש צורך לתת ערכים לאלמנט אשר לא ישפיעו על התנהגותו, אך דרכם נוכל לקבל מידע על האלמנט. לדוגמא:

נציב על הדף 5 לחצנים ו-5 תיבות טקסט. בלחיצה על כל לחצן יופיע הכיתוב שעליו בתיבת הטקסט שלידו. כיצד נדע מי היא תיבת הטקסט שליד הלחצן? לא ניתן לתת לשתי הפקדים ערך זהה בתכונה `id` כיון שזהו ערך יחודי. אם כן, כיצד נקשר ביניהם? ע"י התכונה `data-`. תכונה זו מאפשרת להוסיף לאלמנט תכונה משלנו, שלא תשפיע על התנהגות האלמנט, אך דרכה ניתן יהיה לדעת על האלמנט מידע נצרך. הערך המוצב בתכונה זו יכול להיות כל מידע שיש בו צורך. השם יתחיל ב-`data-` ואח"כ כל שם שעונה על כללי שמות משתנים (ללא רווח וכו').

```
<input type="text" data-myButton="btn1" />  
<input type="button" id="btn1" />
```

ניתן לתת ערך למאפיין גם בזמן ריצה:

```
var txt = document.createElement("input");  
txt.setAttribute("type", "text");  
txt.setAttribute("data-myAttribute", "stam")
```

תרגיל מס' 4

צרי פאזל בגודל 3X3

הכיני תמונות לפאזל ע"י חלוקת תמונה גדולה בעורך תמונות גרפי.

שמרי את כל התמונות בפרויקט.

למשתמש יוצגו רשימת התמונות בשורה וטבלה לבניית הפאזל.

הטבלה תהיה פנויה עם קווי מסגרת (`border=1`). לכל תא בטבלה ניתן לגרור תמונה.

שם האלמנט יהיה מתאים לשם התא בו הוא ימוקם, לדוגמא:

שם התמונה הראשונה: `img1` שם התא הראשון בטבלה: `td1`

בעת גרירת האלמנט יש לבדוק שהוא נגרר לתא המתאים, ע"י התאמת המס' הסידורי.

בסיום הרכבת הפאזל, השמיעי צליל של מחיאת כפיים.

