

JQuery

jQuery

תוכן עניינים

3	1. הקדמה.....
3	1.1. יתרונות שימוש ב- jQuery.....
4	1.2. מה צריך לדעת כדי ללמוד jQuery ?
4	1.3. הורדה של jQuery.....
4	1.4. דוגמא ראשונה לעבודה עם jQuery.....
6	2. הרצת Script בזמן טעינה של הדף.....
7	3. בחירת אלמנטים.....
13	4. גישה לאלמנטים ב- DOM.....
17	5. שינוי תכונות של אלמנטים.....
19	5.1. שיוך class לאלמנטים.....
20	5.2. עיצוב אלמנטים.....
21	6. עדכון תוכן של אלמנט.....
24	7. הוספת אלמנטים בצורה דינמית לדף.....
27	8. רישום לאירועים.....
32	9. אנימציות/אפקטים.....
34	10. שרשור אירועים.....
34	11. יצירת accordion באמצעות jQuery.....
38	12. AJAX.....
38	12.1. פונקציית \$ajax.....
40	12.2. פונקציית \$ajaxSetup.....
41	12.3. get.....
42	12.4. post.....
42	12.5. load.....
43	12.6. JQuery AJAX Events.....
44	13. JQuery Plugins.....
48	14. JQuery UI.....
51	15. כתיבת Custom Plugins.....
57	16. תרגילים.....
64	17. פתרונות לתרגילים נבחרים.....
68	18. מקורות לקריאה נוספת.....

1. הקדמה

jQuery הינה ספריה חנימית של JavaScript המפשטת את העבודה עם מסמכי HTML ו-CSS. jQuery היא תוכנת קוד פתוח הפועלת תחת רישיון MIT. ספריה זו פותחה בשנת 2006 על-ידי John Resig, והגרסה האחרונה שלה היא 3.2.1 שיצאה במרץ 2017.

jQuery משולבת בפלטפורמות שונות של חברות עולמיות כמו Microsoft ו-Samsung (משלבת ב Visual Studio את הספריות של jQuery בתוך ה- Templates לבניית אתרים של ASP.NET, MVC ו-), ועוד.

jQuery מספקת לנו את היכולות הבאות:

- בחירה של אלמנט או קבוצת אלמנטים מתוך דף html
 - שינוי אלמנט / קבוצת אלמנטים ע"י שינוי הגדרות ה-CSS
 - הוספת אלמנטים חדשים לדף
 - מחיקת אלמנטים מהדף
 - הוספת פונקציונליות AJAX לדף
 - הגדרת פונקציות כתגובה לאירועים שונים הפועלים על אלמנט מסוים (או קבוצת אלמנטים)
 - אנימציה ואפקטים
- כמו-כן, קיימות הרחבות רבות לספריה jQuery הבסיסית שמספקות רכיבים נוספים כמו תפריטים מורכבים, גלרית תמונות, active grid – פקד המאפשר להציג מידע טבלאי ולערוך אותו במקום ועוד.

1.1. יתרונות שימוש ב- jQuery

- גודל הקוד של הספריות קטן
- ביצועים טובים של הקוד
- כמות עשירה של תוספות
- תיעוד מפורט ובהיר
- חוסך זמן בכתיבת קוד JavaScript
- הפרדת מבנה ועיצוב הדף מהלוגיקה והטיפול באירועים

1.2. מה צריך לדעת כדי ללמוד jQuery ?

דרוש ידע בשפות HTML, CSS ו-JavaScript על-מנת ללמוד jQuery.

1.3. הורדה של jQuery

הגרסה האחרונה של jQuery ניתנת להורדה מדף הבית של jQuery בכתובת:

<http://www.jquery.com>

מומלץ לשים את הספריות של jQuery בתת-ספריה כלשהי באתר שאתם בונים (למשל בתת-ספריה בשם Scripts), על-מנת שניתן יהיה להתייחס אליהן בקוד HTML.

1.4. דוגמא ראשונה לעבודה עם jQuery

נראה כיצד רושמים "Hello World" בתוך אלמנט <div> בדף HTML באמצעות JQuery.

ניצור דף HTML חדש בשם default.html.

על-מנת לעבוד עם הספריות של jQuery, יש להוסיף אליהן קישור בדף באמצעות תגית ה- <script> הבאה (שורה זו תשתנה בהתאם לגרסת jQuery שנמצאת אצלכם במחשב, ובהתאם למיקום של הספריה אצלכם במחשב).

```
<script src="Scripts/jquery-3.2.1.js" type="text/javascript"></script>
```

נוסיף את ה- <div> הבא לדף HTML:

```
<div id="div1" class="classic">  
  I am just a boring div element  
</div>
```

וניתן לו עיצוב בסיסי באמצעות CSS:

```
<style type="text/css">  
  .classic  
  {  
    background: #BBBBBB;  
    width: 250px;  
    height: 100px;  
  }  
</style>
```

עתה בראש הדף נוסיף קוד סקריפט שיכיל קריאה לפונקציה בסיסית של jQuery שתשנה את התוכן של ה- <div>:

```
<script type="text/javascript">  
  function hellojQuery() {
```

```

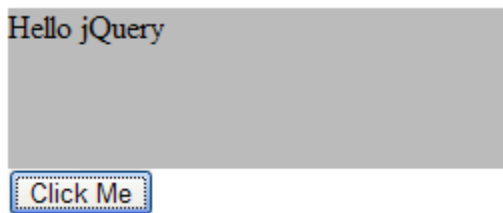
        $("#div1").html("Hello jQuery");
    }
</script>

```

ועתה נוסיף לחצן שיפעיל את הפונקציה שכתבנו:

```
<input type="button" value="Click Me" onclick="hellojQuery();" />
```

התוצאה לאחר הלחיצה על הלחצן היא:



עתה ננתח את שורת הקוד שרשמנו:

```
$("#div1").html("Hello jQuery");
```

- הסימן \$ הוא קיצור עבור הדפדפן האומר לו להשתמש במרחב השמות של jQuery. אפשר לרשום במקום \$ גם jQuery("#div1") אבל צורת כתיבה זו ארוכה יותר ולכן פחות מומלצת.
- הטקסט "#div1" מציין ל-JQuery לבחור את כל האלמנטים בדף שיש להם ID בשם div1, בדומה לשימוש ב-Selector-ים ב-CSS. כתוצאה מכך, jQuery מחזיר את קבוצת כל האלמנטים בדף התואמים ל-Selector. שימו לב שגם אם ה-Selector מחזיר רק אלמנט אחד, עדיין התוצאה תהיה קבוצה של אלמנטים. (כלומר האלמנטים שנבחרו תמיד יחזרו במערך, שאורכו יקבע על פי כמות האלמנטים שנבחרו, במקרה ונבחר רק אלמנט אחד – אורך המערך יהיה 1).
- jQuery תומך בכל ה-Selector-ים הקיימים ב-CSS, וגם מוסיף כמה משלו, שנלמד עליהם בהמשך. הפונקציה **html()** היא גרסת jQuery למאפיין innerHTML (של javascript) ומאפשרת לבצע שינוי בתוכן של כל האלמנטים בקבוצה שנבחרה קודם לכן.

2. הרצת Script בזמן טעינה של הדף

ישנן פעולות מסוימות שנרצה לבצע בזמן הטעינה של הדף. ב-JavaScript היינו כותבים את הפעולות האלה באירוע `onload()` של החלון. הבעיה באירוע הזה היא שהוא מתרחש לאחר שכל הדף נטען, כולל כל התמונות שבתוכו. jQuery מציעה פתרון עדיף על-ידי שימוש באירוע `ready` של המסמך. אירוע זה מתרחש לאחר טעינת האלמנטים של הדף אבל טרם טעינת התמונות.

לדוגמא, נוכל להציג הודעת `alert` בזמן הטעינה של הדף באופן הבא:

```
<script type="text/javascript">
  $(document).ready(function () {
    alert("document is ready");
  });
</script>
```

שימו לב שהפונקציה `$(document).ready()` מקבלת בתור פרמטר פונקציה אחרת שתופעל לאחר שהדף נטען.

מרבית הקוד בדף שנכתוב ב-JQuery ייכתב בתוך אירוע ה-`document.ready`.

ניתן לקצר ובמקום לרשום `$(document).ready(...)` לרשום `$(...)`, למשל את אותו סקריפט ניתן לרשום גם באופן הבא:

```
<script type="text/javascript">
  $(function () {
    alert("document is ready");
  });
</script>
```

בד"כ נשתמש בכתיבה המקוצרת שהיא יותר נוחה.

3. בחירת אלמנטים

אחד היתרונות של jQuery הוא הקלות שבה ניתן לבחור אלמנטים בדף ולבצע עליהם מניפולציות שונות. jQuery תומכת ב-Selector-ים הקיימים ב-CSS ובנוסף מציעה סלקטורים שונים שלא נתמכים ב-CSS.

סלקטורי CSS בסיסיים:

סלקטור	דוגמא לשימוש	משמעות הדוגמא
All Selector ("*")	\$("*")	בחירת כל האלמנטים בעמוד, כולל body וה-head
ID Selector ("#id")	\$("#div1")	בחירת האלמנט עם המזהה div1
Element Selector("element")	\$("div")	בחירת כל ה-div-ים בדף
Class Selector (".class")	\$(".classic")	בחירת כל האלמנטים השייכים למחלקה classic
Multiple Selector ("selector1, selector2, selectorN")	\$(".classic, #div1")	בחירת כל האלמנטים השייכים למחלקה classic או עם המזהה div1
Child Selector ("parent > child")	\$(".classic > #div1")	בחירת כל אלמנט עם המזהה div1 הנמצא בתוך אלמנט השייך למחלקה classic
Descendant Selector ("ancestor descendant")	\$("#div1 #div2")	בחירת האלמנט עם מזהה div2 שנמצא בתוך אלמנט עם מזהה div1
Next Adjacent Selector ("prev + next")	\$("#div1 + #div2")	בחירת האלמנט עם מזהה div2 המופיע מיד לאחר אלמנט עם מזהה div1
Next Siblings ("prev ~ siblings")	\$("#div1 ~ #div2")	בחירת האלמנט עם מזהה div2 המופיע לאחר אלמנט עם מזהה div1

סלקטורי פילטור לפי מיקום:

סלקטור	דוגמא לשימוש	משמעות הדוגמא
:first Selector	<code>\$("div:first")</code>	בחירת ה-div הראשון בדף
:last Selector	<code>\$("div:last")</code>	בחירת ה-div האחרון בדף
:first-child Selector	<code>\$("div:first-child")</code>	בחירת אלמנט הבן הראשון
:last-child Selector	<code>\$("div:last-child")</code>	בחירת אלמנט הבן האחרון
: first-of-type Selector	<code>\$("div:fist-of-type")</code>	בחירת כל אלמנט מסוג div שהוא הראשון בקבוצת ה siblings בה הוא ממוקם
:last-of-type Selector	<code>\$("div:last- of-type")</code>	בחירת כל אלמנט מסוג div שהוא האחרון בקבוצת ה siblings בה הוא ממוקם
:eq() Selector	<code>\$("li:eq(2)")</code>	בחירת ה-li השלישי (שנמצא באינדקס 2)
:even Selector	<code>\$("div:even")</code>	בחירת ה-div ימים במקומות הזוגיים
:odd Selector	<code>\$("div:odd")</code>	בחירת ה-div ימים במקומות האי-זוגיים
:gt() Selector	<code>\$(" div:gt(3)")</code>	בחירת ה-div ימים בעלי אינדקס גדול מ-3
:lt() Selector	<code>\$(" div:lt(3)")</code>	בחירת ה-div ימים בעלי אינדקס קטן מ-3
:root Selector	<code>\$(" :root")</code>	בוחר את אלמנט ה-html

בחירת כל אלמנטי ה-div שהם שלוש לפני הסוף	<code>\$("div:nth-last-child(3)")</code>	:nth-last-child() Selector
בחירת כל אלמנטי ה-div שהם הבן השלישי	<code>\$("div:nth-child(3)")</code>	:nth-child() Selector
בחירת כל אלמנטי ה-div שהם בן יחיד	<code>\$("div:only-child")</code>	:only-child() Selector

סלקטורי פילטור לפי תוכן:

משמעות הדוגמא	דוגמא לשימוש	סלקטור
בחירת כל ה-div-ים המכילים את התוכן "JBH"	<code>\$("div:contains('JBH') ")</code>	:contains() Selector
בחירת כל ה-div-ים שלא מכילים טקסט או child elements	<code>\$("div:empty")</code>	:empty Selector
בחירת כל ה-div-ים המכילים child element אחד לפחות (מסוג a)	<code>"div:has(a)"</code>	:has() Selector
בחירת כל ה-div-ים שמכילים child element אחד לפחות	<code>\$("div:parent")</code>	:parent Selector

סלקטורי פילטור מתקדמים:

סלקטור	דוגמא לשימוש	משמעות הדוגמא
:hidden Selector	<code>\$("div:hidden")</code>	בחירת ה-divים שלא מוצגים בדף (לדוגמא: div בעל רוחב וגובה 0, div בעל הגדרת CSS של display:none וכו')
:visible Selector	<code>\$("div:visible")</code>	בחירת ה-divים התופסים מקום בדף
language Selector	<code>\$("div:lang(en-us)")</code>	בחירת ה-divים המכילים את attribute: lang="en-us"
Attribute Starts With	<code>\$("a[href^=http://]")</code>	בחירת כל הלינקים לכתובות שמתחילות ב-http://
Attribute Contains Prefix Selector	<code>\$("a[href= http]")</code>	בחירת כל הלינקים לכתובות שמכילות רק http:// או מכילות בהתחלה http:// ומיד אחר כך את הסימן -
Attribute Contains Selector	<code>\$("a[href*= http]")</code>	בחירת כל הלינקים לכתובות שמכילות בכתובת את http://
Attribute Contains Word Selector	<code>\$("a[href~= jbh.co.il]")</code>	בחירת כל הלינקים לכתובות שמכילות רק את התוכן jbh.co.il או שמכילות את jbh.co.il בתור מילה נפרדת
Attribute Ends With Selector	<code>\$("a[href\$.zip]")</code>	בחירת כל הלינקים לקבצים שמסתיימים בסיומת zip
Attribute Equals Selector	<code>\$("input[type=button]")</code>	בחירת כל הפקדים מטיפוס button

בחירת כל הפקדים שלא מטיפוס button	\$("input[type!=button]")	Attribute Not Equals Selector
בחירת ה-divים שיש להם id	\$("div[id]")	Has Attribute Selector
בחירת ה-headerים בדף (header1-header6)	\$(":header")	:header Selector
בחירת כל ה-divים שלא שייכים למחלקה classic	\$("div:not(.classic)")	:not() Selector
בחירת כל ה-divים שנמצאים במצב אנימציה	\$("div:animated")	:animated Selector

סלקטורי form (לא קיימים ב-CSS):

סלקטור	דוגמא לשימוש	משמעות הדוגמא
:input Selector	\$(":input")	בחירת כל הלחצנים ותיבות הקלט בדף
:button Selector	\$(":button")	בחירת כל הלחצנים בדף
:checkbox Selector	\$(":checkbox")	בחירת כל ה-checkbox-ים בדף
:file Selector	\$(":file")	בחירת כל הפקדים מסוג input type="file"
:image Selector	\$(":image")	בחירת כל הפקדים מסוג input type="image"
:password Selector	\$(":password")	בחירת כל הפקדים מסוג input type="password"
:radio Selector	\$(":radio")	בחירת כל הפקדים מסוג input type="radio"
:reset Selector	\$(":reset")	בחירת כל הפקדים מסוג input type="reset"
:submit Selector	\$(":submit")	בחירת כל הפקדים מסוג input type="submit"
:text Selector	\$(":text")	בחירת כל הפקדים מסוג input type="text"
:selected Selector	\$(":selected")	בחירת כל אלמנטי option שנבחרו (option הוא child של פקד select)
:checked Selector	\$(":checked")	בחירת כל ה-checkbox-ים המסומנים
:enabled Selector	\$(":enabled")	בחירת כל הפקדים שבמצב enabled
:disabled Selector	\$(":disabled")	בחירת כל הפקדים שמצב disabled
:focus Selector	\$(":focus")	בחירת הפקד שהפוקוס עליו

4. גישה לאלמנטים ב-DOM

ישנן מספר פונקציות ב-jquery המסייעות לטייל בעץ של ה-DOM:

parent()	בחירת אלמנט האב
children()	בחירת כל האלמנטים הבנים של האלמנט הנוכחי. ניתן לתת כפרמטר לפונקציה selector, שיבחר רק את הבנים של התגית העונים על התנאי של ה-selector.
siblings()	בחירת כל האחים של האלמנט הנוכחי
next()	בחירת האח הבא של האלמנט הנוכחי
nextAll()	בחירת כל האחים הבאים של האלמנט הנוכחי
prev()	בחירת האח הקודם של האלמנט הנוכחי
prevAll()	בחירת כל האחים הקודמים של האלמנט הנוכחי

אופן השימוש הבסיסי בפונקציות הוא ע"י התחביר הבא:

```
$(תוקן)(שם הפונקציה הרצויה);  
הסלקטור הסלקטור
```

לדוגמא, נניח שנרצה להציג את מספר הלינקים הנמצאים בתוך הפסקה הבאה:

```
<p id="p1">  
  This is a first link: <a href="page1.html">Page1</a><br />  
  This is a second link: <a href="page2.html">Page2</a><br />  
  This is a third link: <a href="page3.html">Page3</a><br />  
</p>
```

נוכל לבצע זאת באמצעות הקוד הבא:

```
alert($('#p1').children('a').size());
```

התוצאה המתקבלת היא 3.

עבודה עם קבוצות אלמנטים

לאחר בחירת קבוצת אלמנטים באמצעות אחד מה-Selector-ים, ניתן לבצע על הקבוצה פעולות שונות. על-מנת להתייחס לאלמנט ספציפי בקבוצה, ניתן להשתמש בסוגריים מרובעים ולציין את מיקומו הסידורי של האלמנט שנרצה להתייחס אליו (בדומה למערכים ב-JavaScript), לדוגמא, כדי להתייחס ל-div הראשון במסמך נוכל לרשום:

```
$("#div")[0]
```

על-מנת לדעת כמה אלמנטים נמצאים בקבוצה ניתן להשתמש בפונקציה size, לדוגמא:

```
$("#div").size()
```

ישנן פעולות שניתן לבצע על הקבוצה והן מופעלות אוטומטית על כל האלמנטים בקבוצה (מנגנון הנקרא **implicit iteration**), לדוגמא כדי לשנות את התוכן של כל ה-div-ים בדף נוכל לרשום:

```
$("#div").html("I am a div");
```

הדבר חוסך לנו כתיבה של הלולאה שתעבור על כל האלמנטים בקבוצה ותשנה את התוכן של כל אחד מהם בנפרד.

אם בכל זאת נרצה לעבור על האלמנטים בקבוצה בזה אחר זה, ולבצע פעולה מסוימת על כל אחד בנפרד, נוכל להשתמש בפונקציה each().

פונקציה זו עובדת בצורה דומה ללולאת for ב-JavaScript.

התחביר הבסיסי שלה הוא:

הפונקציה שתבצע על כל אלמנט מקבוצת האלמנטים שנבחרו תוכן הסלקטור

```
$( ).each(function (index , element) { } );
```

הפרמטרים המועברים לפונקציה הם:

(1) **אינדקס** – מטיפוס מספרי, המייצג את מיקומו של האלמנט הנוכחי במערך

(2) **אלמנט** – תוכן האלמנט הנוכחי עליו מבוצעת האיטרציה

שימו לב לצורת הפעלת הפונקציה each() – הפונקציה הזו מקבלת בתור פרמטר פונקציה אחרת שתופעל על כל אחד מהאלמנטים בקבוצה שנבחרה.

הערה: הרבה מהפונקציות ב-JQuery מקבלות בתור פרמטר פונקציות אחרות של JavaScript המשמשות אותן לצורך ביצוע פעולות מסוימות (מנגנון זה נקרא **callback functions**).

נקודות חשובות בנוגע לשימוש בפונקציית each:

1. בשימוש בפונקציה, אין חובה להשתמש בכל הפרמטרים אם אין בהם צורך, ולכן אפשר להשתמש בפונקציה `each` גם בצורות הבאות:

<p>הפונקציה שתבוצע על כל אלמנט מקבוצת האלמנטים שנבחרו</p> <p>תוכן הסלקטור</p> <pre>\$().each(function () { });</pre>	בלי פרמטרים בכלל
<p>הפונקציה שתבוצע על כל אלמנט מקבוצת האלמנטים שנבחרו</p> <p>תוכן הסלקטור</p> <pre>\$().each(function (index) { });</pre>	עם האינדקס בלבד

2. אם נרצה להפסיק את ביצוע פונקציית ה-callback על האיטרציות הבאות, נוכל לעשות זאת באמצעות הוספת השורה

return false;

בדוגמאות הבאות נשתמש בפונקציה `html()`. השייכת לספריית jquery. הפונקציה `html()` מעדכנת את התוכן ה-HTML של האלמנט (בדומה למאפיין `innerHTML` של JavaScript).

דוגמא:

ניצור דף `html` המכיל בחלק `body` רק את התוכן הבא:

```
<div></div>
<div></div>
<div></div>
```

נכתוב פונקציה שתעבור על כל ה-div-ים ותשנה את התכונה `html` של כל `div` כך שהיא תציג את מיקומי הסידורי של ה-div הזה בדף:

```
$(function() {
  $("div").each(function(index) {
    $(this).html("I am div number " + index);
  });
});
```

תוצאת הפעלת הפונקציה בזמן טעינת העמוד:

I am div number 0

I am div number 1

I am div number 2

דוגמא נוספת:

ניצור דף html המכיל בחלק body רק את התוכן הבא:

```
<div>John Bryce</div>  
<div>Tel Aviv</div>  
<div>Jquery</div>
```

נכתוב פונקציה שתעבור על כל ה-div-ים ותשנה את התכונה html של ה-div הראשון שתוכנו "Tel Aviv" (כאשר תמצא div מתאים, היא תפסיק לרוץ על שאר האלמנטים, כי כבר ביצעה את הפעולה הרצויה):

```
$(function() {  
    $("div").each(function(index, element) {  
        if ($(element).html() == "Tel Aviv") {  
            $(element).html("This is the wanted div")  
            return false;  
        }  
    });  
});
```

תוצאת הפעלת הפונקציה:

John Bryce
This is the wanted div
Jquery

5. שינוי תכונות של אלמנטים

jQuery מספקת פונקציות לקריאה ושינוי של תכונות של אלמנטים בדף.

קריאת ערך attribute	עריכת ערך attribute
<pre>\$.attr();</pre> <p>בתוך הסוגריים הריקים נרשום את תוכן הסלקטור לקרוא</p>	<pre>\$.attr(, ,);</pre> <p>הערך החדש שנרצה להכניס שם attribute אותו נרצה לשנות</p>

לדוגמא, על-מנת להציג את המזהה של האלמנט div1 בדף נוכל לרשום:

```
alert($("#div1").attr("id"));
```

על-מנת לשנות את התכונה title של div1 (תכונה זו קובעת את ה- tooltip שמוצג כאשר העכבר מרחף מעל האלמנט) נוכל לרשום:

```
$("#div1").attr("title", "hello");
```

בנוסף, נוכל גם לשנות ערכים לכמה attributes בפעם אחת, כמו שנראה בדוגמא הבאה:

```

<!DOCTYPE html>

<html lang="en">
<head>
<title>JBH JQuery</title>
<script type="text/javascript" src="/scripts/jquery-3.2.1.js">
</script>

<script type="text/javascript" language="javascript">

    $(function() {

        $("a").attr({
            href: "https://api.jquery.com/",
            title: "שיעור ב - jquery",
            style: "color:red;"
        });

    });

</script>
</head>
<body>
<a > my link </a>

</body>
</html>

```

התוצאה של הדף שנקבל (כאשר נעמוד עם הסמן על הלינק):

my link

שיעור ב - jquery

הערה: ניתן להסיר attribute מאלמנט מסוים, באמצעות הפונקציה **removeAttr**.



לדוגמא, על- מנת להסיר את ה- attribute של title מהאלמנט div1 בדף נוכל לרשום:

```
$("#div").removeAttr("title ");
```

5.1. שיוך class לאלמנטים

- ניתן להוסיף או להסיר שיוך של אלמנט למחלקה מסוימת באמצעות הפונקציות addClass() ו-removeClass(), לדוגמא כדי להוסיף את המחלקה specialDiv לכל ה-divים בדף נוכל לרשום:

```
$("#div").addClass("specialDiv");
```

ובכדי להסיר את המחלקה specialDiv לכל ה-divים בדף נוכל לרשום:

```
$("#div").removeClass("specialDiv");
```

- ניתן באמצעות פקודה אחת (הפקודה toggleClass) להוסיף שיוך של אלמנט למחלקה מסוימת אם המחלקה לא משוייכת לאלמנט, או להסיר שיוך של אלמנט למחלקה מסוימת אם היא כבר משוייכת לאלמנט:

לדוגמא, כאשר נרשום את הפקודה הבאה:

```
$("#div").toggleClass("specialDiv");
```

במקרה בו המחלקה specialDiv משוייכת לכל ה-divים בדף היא תוסר, ובמקרה בו המחלקה specialDiv לא משוייכת לכל ה-divים בדף היא תשוויר.

- ניתן לבדוק האם אלמנט משוייך למחלקה מסוימת באמצעות הפונקציה hasClass(), שמחזירה ערך בוליאני בהתאם.

לדוגמא כדי לבדוק האם המחלקה specialDiv משוייכת לכל ה-divים בדף, ולהדפיס הודעה מתאימה בהתאם, נוכל לרשום:

```
if($("#div").hasClass("specialDiv")){
    alert("Every div has the class specialDiv");
}
else {
    alert("Not every div has the class specialDiv");
}
```

5.2. עיצוב אלמנטים

ב- jquery נוכל לשנות את ערכי css בצורה קלה והוחה, באמצעות הפונקציה `css()`:

קריאת ערך	עריכת ערך	
פרמטר בודד	לדוגמא: קבלת צבע הרקע של אלמנטי p <code>\$("#p").css("background-color");</code>	לדוגמא: שינוי צבע הרקע לאלמנטי p <code>\$("#p").css("background-color", "yellow");</code>
פרמטרים מרובים	לדוגמא: קבלת רוחב, אורך וצבע רקע של אלמנטי p <code>\$("#p").css(["width", "height", "background-color"]);</code>	לדוגמא: עריכת רוחב, אורך וצבע רקע לאלמנטי p <code>\$("#p").css({backgroundColor: "black", color: "white"});</code>

מאפייני עיצוב נוספים הניתנים לשינוי באמצעות פונקציות יעודיות ב- jquery:

הערך	משמעות	דוגמא לקריאת הערך	דוגמא לעריכת הערך
<code>height()</code>	הגובה המחושב ב- css	<code>\$("#p").height();</code>	<code>\$("#p").height(100);</code>
<code>innerHeight()</code>	הגובה הפנימי (כולל padding אך לא border)	<code>\$("#p").innerHeight();</code>	<code>\$("#p").innerHeight(130);</code>
<code>outerHeight()</code>	הגובה החיצוני (כולל padding ו- border)	<code>\$("#p").outerHeight();</code>	<code>\$("#p").outerHeight(150);</code>
<code>width()</code>	הרוחב המחושב ב- css	<code>\$("#p").width();</code>	<code>\$("#p").width(100);</code>
<code>innerWidth()</code>	הרוחב הפנימי (כולל padding אך לא border)	<code>\$("#p").innerWidth();</code>	<code>\$("#p").innerWidth(130);</code>
<code>outerWidth()</code>	הרוחב החיצוני (כולל padding ו- border)	<code>\$("#p").outerWidth();</code>	<code>\$("#p").outerWidth(150);</code>
<code>offset()</code>	ה- coordinates הנוכחים של האלמנט, יחסית ל- document.	<code>var coord = \$("#p").offset();</code> <code>var a = coord.left;</code> <code>var b = coord.top;</code>	<code>\$("#p").offset({ top: 10, left: 30 });</code>

<code>\$("#p").scrollLeft();</code>	<code>\$("#p").scrollLeft(300);</code>	המיקום האופקי הנוכחי של ה- scroll bar עבור האלמנט	scrollLeft()
<code>\$("#p").scrollTop(300);</code>	<code>\$("#p").scrollTop();</code>	המיקום האנכי הנוכחי של ה- scroll bar עבור האלמנט	scrollTop()

6. עדכון תוכן של אלמנט

ניתן לעדכן תוכן של אלמנט בדף באמצעות אחת מהפונקציות:

1) `html()` - מעדכנת את התוכן ה-HTML של האלמנט (בדומה למאפיין `innerHTML` של JavaScript)

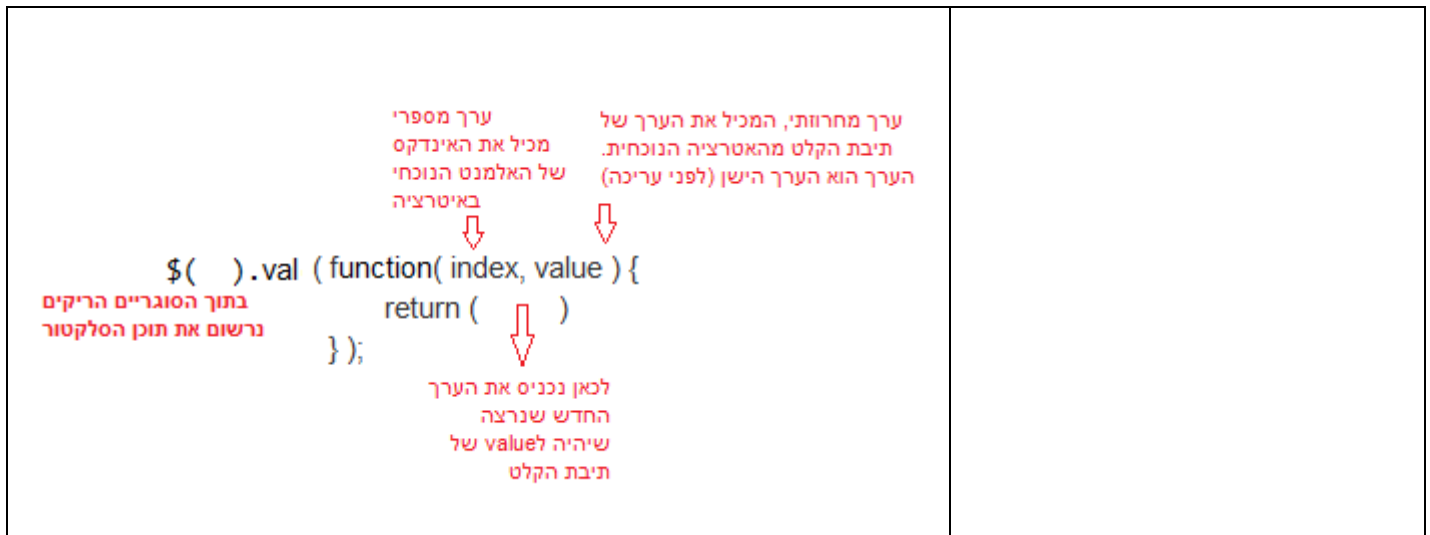
קריאת תוכן ה- html מאלמנט	עריכת תוכן ה- html לאלמנט
<pre>\$().html();</pre> <p>· בתוך הסוגריים הריקים נרשום את תוכן הסלקטור</p>	<p>אופציה ראשונה: עריכה ע"י שליחת מחרוזת</p> <pre>\$().html();</pre> <p>· בתוך הסוגריים הריקים נרשום את תוכן הסלקטור</p> <p>הערך החדש שנרצה להכניס</p> <p>אופציה שניה: עריכה על ידי שליחת פונקציה כפרמטר (הפונקציה שנשלחת כפרמטר, מחזירה מחרוזת)</p> <pre>\$().html(function(index, value) { return (); });</pre> <p>· בתוך הסוגריים הריקים נרשום את תוכן הסלקטור</p> <p>ערך מספרי מכיל את האינדקס של האלמנט הנוכחי באיטרציה</p> <p>ערך מחרוזתי, המכיל את התוכן של האלמנט הנוכחי באיטרציה</p> <p>לכאן נכניס את הערך החדש שנרצה</p>

(2) `.text()` – עדכון התוכן המילולי של האלמנט (בדומה למאפיין `innerText` של JavaScript).

קריאת תוכן ה- <code>text</code> מאלמנט	עריכת תוכן ה- <code>html</code> לאלמנט
<p><code>\$().text()</code></p> <p>בתוך הסוגריים הריקים נרשום את תוכן הסלקטור</p>	<p>אופציה ראשונה: עריכה ע"י שליחת מחרוזת</p> <p><code>\$().text(↓);</code></p> <p>הערך החדש שנרצה להכניס</p> <p>בתוך הסוגריים הריקים נרשום את תוכן הסלקטור</p> <p>אופציה שנייה: עריכה על ידי שליחת פונקציה כפרמטר (הפונקציה שנשלחת כפרמטר, מחזירה מחרוזת)</p> <p>ערך מחרוזתי, המכיל את התוכן <code>text</code> של האלמנט באיטרציה הנוכחית</p> <p>ערך מספרי מכיל את האינדקס של האלמנט הנוכחי באיטרציה</p> <p><code>\$().text(function(index, value) { return (↓); });</code></p> <p>בתוך הסוגריים הריקים נרשום את תוכן הסלקטור</p> <p>לכאן נכניס את הערך החדש שנרצה</p>

(3) `.val()` - עדכון התוכן המילולי של אלמנטי `input`

קריאת תוכן תיבת הקלט	עריכת תוכן תיבת הקלט
<p><code>\$().val();</code></p> <p>בתוך הסוגריים הריקים נרשום את תוכן הסלקטור</p>	<p>אופציה ראשונה: עריכה ע"י שליחת מחרוזת / מספר כפרמטר</p> <p><code>\$().val(↓);</code></p> <p>הערך החדש שנרצה להכניס</p> <p>בתוך הסוגריים הריקים נרשום את תוכן הסלקטור</p> <p>אופציה שנייה: עריכה על ידי שליחת פונקציה כפרמטר (הפונקציה שנשלחת כפרמטר, מחזירה מחרוזת)</p>



דוגמא:

ניצור דף html המכיל בחלק ה body שלו רק את התוכן הבא:

```
<input type="text" value="hello"/>  
<input type="text" value="JBH School"/>
```

בתוך הארוע של document.ready נוסיף את הקוד הבא:

```
$( "input[type=text]" ).val(function( index, value ) {  
    return value.toUpperCase();  
});
```

הקוד הזה, שולח פונקציה כפרמטר בקריאה ל- val().
כאשר val() תתבצע, הפונקציה אותה שלחנו כפרמטר תרוץ על כל אלמנט מסוג תיבת טקסט (input type="text") ותקבל את הערך של תיבת הטקסט הנוכחית.
את הערך הזה תהפוך לאותיות גדולות, ותגדיר אותו בתור הערך המעודכן של תיבת הטקסט.

ולכן, כאשר נריץ את התוכנית, נקבל את התוצאה הבאה:

HELLO

JBH SCHOOL

7. הוספת אלמנטים בצורה דינמית לדף

jQuery מאפשרת להוסיף אלמנטים לדף בצורה נוחה. קיימות מספר פונקציות המאפשרות לעשות זאת. בטבלה הבאה, נסקור מספר פונקציות נפוצות לביצוע מניפולציות על עץ ה DOM, כאשר כל הדוגמאות מתבססות על תבנית הדף הבאה:

```
<html lang="en">

<head>
  <script src="scripts/jquery-3.2.1.js"></script>

  <div style="border: 2px solid black; padding: 10px; text-align: center; width: fit-content; margin: 20px auto;">
    לכאן נכניס את קטעי הקוד שנוסיף עבור כל דוגמא
  </div>

</head>

<body>
  <div style="background-color: yellow; id="div1">תוכן סטטי</div>
</body>
```

התוצאה שתוצג לדפדפן (לפני השינויים שנבצע באמצעות jQuery):

תוכן סטטי

כעת, נבצע שינויי אלמנטים בהרכב ה DOM:

תוצאה	דוגמה לשימוש בפונקציה	הפונקציה
<div>תוכן סטטי</div> <div>תוכן דינמי</div>	<pre>\$(function() { \$("#div1") .after("<p style='color:red'>דינמי</p>"); });</pre>	<p>הפונקציה after</p> <p>מאפשרת להוסיף אלמנט חדש לאחר אלמנט אחר.</p> <p>האלמנט החדש יתווסף לאחר התגית.</p>

<div>תוכן סטטי</div> <div>תוכן דינמי</div>	<pre>\$(function() { \$("#div1") .append("<p style='color:red'>דינמי</p>"); });</pre>	הפונקציה append מאפשרת להוסיף אלמנט חדש כאלמנט בן של תגית אחרת. האלמנט החדש יהיה אלמנט הבן האחרון של התגית.
<div>תוכן סטטי</div> <div>תוכן דינמי</div>	<pre>\$(function() { \$("<p style='color:red'>דינמי</p>") .appendTo("#div1"); });</pre>	הפונקציה appendTo מאפשרת להוסיף אלמנט חדש כאלמנט בן של תגית אחרת. האלמנט החדש יהיה אלמנט הבן האחרון של התגית.
<div>תוכן דינמי</div> <div>תוכן סטטי</div>	<pre>\$(function() { \$("#div1") .before("<p style='color:red'>דינמי</p>"); });</pre>	הפונקציה before מאפשרת להוסיף אלמנט חדש לפני אלמנט אחר. האלמנט החדש יתווסף לפני התגית.
	<pre>\$(function() { \$("#div1") .empty(); });</pre>	הפונקציה empty מאפשרת להסיר את כל התוכן והבנים מתוך האלמנט.
<div>תוכן סטטי</div> <div>תוכן דינמי</div>	<pre>\$(function() { \$("<p style='color:red'>דינמי</p>") .insertAfter("#div1"); });</pre>	הפונקציה insertAfter מאפשרת להוסיף אלמנט חדש לאחר אלמנט אחר. האלמנט החדש יתווסף לאחר התגית.
<div>תוכן דינמי</div> <div>תוכן סטטי</div>	<pre>\$(function() { \$("<p style='color:red'>דינמי</p>") .insertBefore("#div1"); });</pre>	הפונקציה insertBefore מאפשרת להוסיף אלמנט חדש לפני אלמנט אחר. האלמנט החדש יתווסף לפני התגית.

<div>תוכן דינמי</div> <div>תוכן סטטי</div>	<pre>\$(function() { \$("#div1") .prepend("<p style='color:red'>דינמי</p>"); });</pre>	הפונקציה prepend מאפשרת להוסיף אלמנט חדש כאלמנט בן של תגית אחרת. האלמנט החדש יהיה אלמנט הבן הראשון של התגית.
<div>תוכן דינמי</div> <div>תוכן סטטי</div>	<pre>\$(function() { \$("<p style='color:red'>דינמי</p>") .prependTo("#div1"); });</pre>	הפונקציה prependTo מאפשרת להוסיף אלמנט חדש כאלמנט בן של תגית אחרת. האלמנט החדש יהיה אלמנט הבן הראשון של התגית.
	<pre>\$(function() { \$("#div1") .remove(); });</pre>	הפונקציה remove מאפשרת להסיר את כל האלמנטים התואמים מ-DOM.
<div>מחליף תוכן</div> <div>תוכן סטטי</div>	<pre>\$(function() { \$("<p style='color:red'>דינמי</p>") .prependTo("#div1"); \$('<div style="color:blue">תוכן מחליף</div>') .replaceAll(\$("p")); });</pre>	הפונקציה replaceAll מאפשרת להחליף אלמנט מסוים באלמנט חדש.
<div>מחליף תוכן</div> <div>תוכן סטטי</div>	<pre>\$(function() { \$("<p style='color:red'>דינמי</p>") .prependTo("#div1"); \$("p").replaceWith ('<div style="color:blue">תוכן מחליף</div>'); });</pre>	הפונקציה replaceWith מאפשרת להחליף אלמנט מסוים באלמנט חדש.

<p>תוכן דינמי</p> <p>תוכן סטטי</p>	<pre><script> \$(function() { \$("#div1").wrap (("p style='color:red'>דינמי תוכן</p>")); }); </script></pre>	<p>הפונקציה wrap</p> <p>מאפשרת לעטוף באלמנט חדש את האלמנט שכבר קיים.</p>
------------------------------------	---	---

8. רישום לאירועים

על-מנת להפריד את העיצוב של הדפים מהקוד שמטפל באירועים, ההמלצה היא לבצע את הרישום לאירועים בקוד JavaScript ולא בתוך תגיות ה-HTML. jQuery מאפשרת להירשם לאירועים של אלמנטים באמצעות הפונקציה **on()**. פונקציה זו מקבלת שני פרמטרים – שם האירוע שאליו רוצים להירשם (יועבר כמחרוזת) וקוד ה-JavaScript שיש להפעיל כאשר האירוע מתרחש (יועבר כפונקציה).

לדוגמא, נוסיף לחצן לדף בשם btnTest, ונרשם לאירוע click שלו בקוד ה-JavaScript:

```
<input type="button" value="Test" id="btnTest" />
$("#btnTest").on("click", function(){
    alert('btnTest was clicked');
});
```

הערה: בגרסאות קודמות היה שימוש בפונקציה bind (זהה לפונקציה on שממומלצת יותר לשימוש בגרסאות החדשות), ולכן ייתכן שנתקל גם בתחביר הבא:

```
<input type="button" value="Test" id="btnTest" />
$("#btnTest").bind('click', function () {
    alert('btnTest was clicked');
});
```



יש לוודא שהרישום לאירוע יבוצע לאחר שהפקד כבר נטען בדף (בהרבה מקרים מבצעים את הרישום לאירועים בתוך האירוע `$(document).ready`).

אם בהמשך הקוד, נרצה לבטל את הפונקציה המתבצעת באירוע, נוכל לעשות זאת באמצעות הפונקציה `.off()`. פונקציה זו מקבלת כפרמטר את שם האירוע שאנו רוצים לבטל (יועבר כמחרוזת).

לדוגמא, נבטל את הפונקציה שהוספנו ללחצן `btnTest` באירוע `click` שלו:

```
$("#btnTest").off("click");
```

או לבטל את כל האירועים של הלחצן `btnTest` באמצעות התחביר:

```
$("#btnTest").off();
```

שם האירוע	הגורם לאירוע
blur	מתרחש כאשר הפוקוס מוסר מהאלמנט
change	מתרחש כאשר האלמנט משתנה
click	מתרחש בעת לחיצה על העכבר
dblclick	מתרחש כאשר בעת לחיצה כפולה (double-click) על העכבר
Focus	מתרחש כאשר האלמנט מקבל פוקוס
keydown	מתרחש בעת הקשה על מקש במקלדת
keypress	מתרחש כאשר מקש הוקש ושוחר
keyup	מתרחש כאשר מקש במקלדת שוחרר
mousedown	מתרחש בעת לחיצה על לחצן העכבר
mouseenter	מתרחש כאשר העכבר נכנס לאזור אלמנט
mouseleave	מתרחש כאשר העכבר עוזב אזור אלמנט
Mousemove	מתרחש כאשר סמן העכבר מוזז
Mouseout	מתרחש כאשר סמן העכבר יוצא מהאלמנט
Mouseover	מתרחש כאשר סמן העכבר עובר על אלמנט
Mouseup	מתרחש כאשר לחצן העכבר משוחרר
Resize	מתרחש כאשר גודל החלון משתנה
Scroll	מתרחש כאשר החלון נגלל
Select	מתרחש בעת בחירת טקסט
Submit	מתרחש כאשר הטופס נשלח
Unload	מתרחש כאשר document במצב unloaded

הערה: בגרסאות קודמות היה שימוש בפונקציה `unbind` (זהה לפונקציה `off` שממומלצת יותר לשימוש בגרסאות החדשות), ולכן ייתכן שנתקל גם בתחביר הבא:

```
$("#btnTest").unbind('click');
```


להלן רשימה של חלק מהאירועים שנוכל ליצור ב-JQuery ע"י bind:

ב-JQuery לאירועים יש קיצורים שמאפשרים להירשם לאירוע על-ידי קריאה לפונקציה הזוהי לשם האירוע, לדוגמא יכולנו לבצע את הרישום לאירוע click גם באמצעות הפונקציה click():

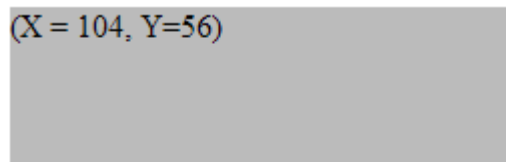
```
$("#btnTest").click(function () {  
    alert('btnTest was clicked');  
});
```

ניתן בנוסף לקבל מידע על האירוע באמצעות שימוש באובייקט **event** (אובייקט שמכיל את הפרמטרים של האירוע שהתבצע).

לדוגמא, נוכל להציג את המיקום של העכבר מעל ה-div בזמן שהמשתמש עובר עם העכבר מעל ל-div באופן הבא:

```
$("#div1").mousemove(function (event) {  
    var str = "(X = " + event.pageX + ", Y=" + event.pageY + ")";  
    $("#div1").text(str);  
});
```


והתוצאה היא:



הפעלה יזומה של אירוע

ב-jquery קיימת הפונקציה trigger המאפשרת לנו לקרוא לאירוע מסויים בצורה יזומה, למרות שלא התרחש בפועל.

התחביר הבסיסי של הפונקציה הוא:

`$()(" ")`
שם האירוע שנרצה להפעיל עבור האלמנט שבחרנו עם הסלקטור
סלקטור

לדוגמא ניצור את העמוד הבא:

```
<html lang="en" xmlns="http://www.w3.org/1999/xhtml">

<head>
  <title>JBH JQuery</title>
  <script src="//ajax.googleapis.com/ajax/libs/jquery/3.2.1/jquery.min.js"></script>
  <script>

    $(function () {

      $("#btnTest").click(function () {
        alert('btnTest was clicked');
      });

      $("#btnTest").trigger("click");

    });

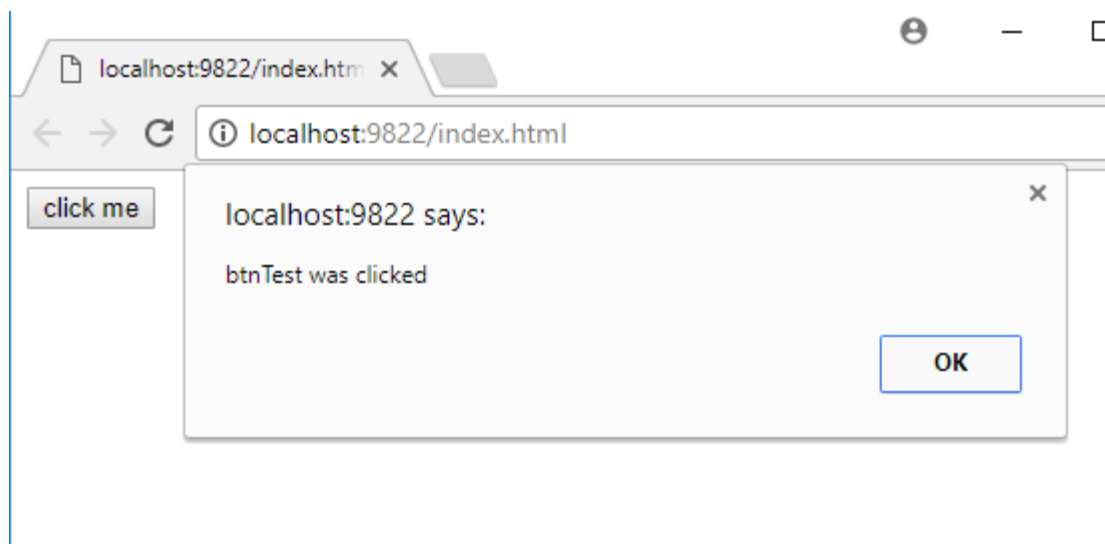
  </script>
</head>

<body>

  <button id="btnTest"> click me </button>

</body>
</html>
```

כאשר נריץ את הדף בדפדפן נקבל את התוצאה הבאה:



הסבר: למרות שלא לחצנו על הכפתור שמפעיל את האירוע הגורם להקפצת הפופאפ, קראנו לו בתוך האירוע של `document.reay` בצורה יזומה ע"י `trigger`, ולכן מיד כשהדף נטען, האירוע הופעל.

9. אנימציות/אפקטים

jQuery מאפשרת להוסיף אנימציות ואפקטים שונים לאלמנטים בדף. לדוגמא, נוכל להוסיף אפקט של fadeout ל- div1 ברגע שלוחצים על הלחצן:

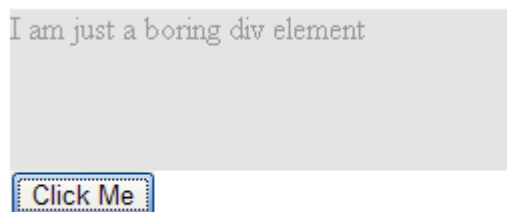
```
function divAnimate() {
    $("#div1").fadeOut();
}


```

ניתן לשלוט בהרבה מהאפקטים על-ידי העברת פרמטרים לפונקציה שמבצעת את האפקט. מרבית האפקטים תומכים בפרמטר הקובע את מהירות הביצוע של האפקט. לדוגמא, ניתן לשלוט במהירות הדעיכה של האלמנט באמצעות פרמטר המציין את משך הדעיכה במילישניות:

```
$("#div1").fadeOut(10000);
```

להלן המראה של הדף אחרי שחלפה כמחצית זמן הדעיכה:



ניתן גם לציין את משך הזמן באמצעות אחת מהמילים 'slow', 'normal' או 'fast', לדוגמא:

```
$("#div1").fadeOut('slow');
```

להלן רשימה של אפקטים נוספים המוגדרים בספריות הסטנדרטיות של jQuery:

\$("#div1").hide()	הסתרת אלמנט
\$("#div1").show()	הצגת אלמנט

<code>\$("#div1").toggle();</code>	מחליף את מצב האלמנט. אם הוא מוסתר – אז הוא יוצג. ואם הוא מוצג- אז הוא יוסתר.
<code>\$("#div1").fadeOut();</code>	fade out
<code>\$("#div1").fadeIn();</code>	fade in
<code>\$("#div1").fadeTo("slow", 0.33);</code>	משנה את ה-Opacity של האלמנט ל-0.33
<code>\$("#div1").fadeToggle();</code>	מחליף את מצב האלמנט. אם הוא מוסתר – אז הוא יוצג. ואם הוא מוצג- אז הוא יוסתר
<code>\$("#div1").slideDown();</code>	אפקט הגורם לאלמנט להופיע בדף תוך כדי גלישה למטה
<code>\$("#div1").slideUp();</code>	אפקט הגורם לאלמנט להופיע בדף תוך כדי גלישה למעלה
<code>\$("#div1").slideToggle();</code>	מחליף את מצב האלמנט. אם הוא מוסתר – אז הוא יופיע בדף תוך כדי גלישה ואם הוא מוצג- אז הוא יוסתר.

בנוסף קיימת פונקציה רבת-עוצמה בשם **animate()** המאפשרת לשנות תכונות CSS של אלמנט מסוים לאורך זמן המוגדר במילישניות.

לדוגמא, נניח שנרצה להגדיל את הרוחב של div1 כך שיתפוס 100% מהדף ולהגדיל את הפונט שלו ל-50px, ולבצע את השינויים האלה במשך שניה וחצי, נוכל לרשום את הקוד הבא:

```
$("#div1").animate({
  width: "100%",
  fontSize: "50px",
}, 1500);
```

ניתן לעצור את האנימציה באמצע באמצעות הפונקציה `stop()`:

```
$("#div1").stop();
```

בנוסף, ניתן גם לבצע delay מסויים בין אפקטים באמצעות הפונקציה `delay()` - המספר שנעביר אליה כפרמטר יקבע את מספר המילישניות שיתבצע ה-delay :

```
$("#div1").slideUp(300).delay(800).fadeIn(400);
```

10. שרשור אירועים

אחד מהיתרונות של jQuery הוא שניתן להפעיל על אותה קבוצת אלמנטים כמה פונקציות בזו אחר זו. לדוגמא, נוכל להפעיל על אותו div כמה אפקטים בזה אחר זה:

```
$("#div1").fadeOut().slideDown().fadeOut().slideUp();
```

11. יצירת accordion באמצעות jQuery

נראה עתה דוגמא שימושית ב-JQuery ליצירת אפקט של accordion בדף (יכול לשמש למשל כדי לבנות רשימה נפתחת של תפריטים).

תחילה נגדיר את התפריטים שלנו ב-HTML. זה מתחת לזה באמצעות תגיות מתאימות, וניתן לכל תפריט כותרת באמצעות תגית <h3> מתאימה:

```
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
  <title>jQuery accordion demo</title>
</head>
<body>
  <h3 class="menu">News</h3>
  <ul class="menuitems">
    <li><a href="http://www.example.com/story1">Story 1</a></li>
    <li><a href="http://www.example.com/story2">Story 2</a></li>
  </ul>

  <h3 class="menu">Articles</h3>
  <ul class="menuitems">
    <li><a href="http://www.example.com/article1">Article 1</a></li>
    <li><a href="http://www.example.com/article2">Article 2</a></li>
    <li><a href="http://www.example.com/article3">Article 3</a></li>
    <li><a href="http://www.example.com/article4">Article 4</a></li>
  </ul>

  <h3 class="menu">Blog Posts</h3>
  <ul class="menuitems">
    <li><a href="http://www.example.com/post1">Post 1</a></li>
    <li><a href="http://www.example.com/post2">Post 2</a></li>
    <li><a href="http://www.example.com/post3">Post 3</a></li>
  </ul>
</body>
</html>
```

```
</ul>
</body>
</html>
```

שימו לב שכל תפריט באקורדיון מורכב משני חלקים:

1. כותרת של התפריט – מוגדרת באמצעות תגית <h3> המשויכת למחלקה menu.
2. הפריטים השייכים לאותו תפריט – נמצאים בתוך תגית המשויכת למחלקה menuitems.

עתה ניתן עיצוב לתפריטים באמצעות CSS מתאים:

```
<style type="text/css">
body
{
    font-family: Geneva,Arial,Helvetica,sans-serif;
}
.menu
{
    width: 200px;
    background-color: purple;
    color: White;
    font-weight: bold;
    padding: 5px;
}
.menuitems
{
    width: 200px;
}
li
{
    list-style-type: none;
}
a
{
    display: block;
    background-color: gray;
    color: White;
    padding: 5px;
    border-bottom: 1px solid white;
    text-decoration: none;
    font-weight: bold;
    margin: 5px;
}
a:hover
```

```
{
  background: lime;
  text-decoration: underline;
}
</style>
```

אפקט האקורדיון ייוצר באמצעות הפונקציה הבאה של JavaScript, אשר תזהה מעבר עם העכבר מעל תפריט כלשהו (באמצעות האירוע mouseover) ובמקרה כזה תבצע שתי פעולות:

1. תפתח את התפריט ותציג את כל הפריטים שלו על-ידי שימוש באפקט ה-`slideDown`.
2. תסגור את כל התפריטים האחרים בדף ותסתיר את הפריטים שלהם על-ידי שימוש באפקט ה-`slideUp`.

הפונקציה תיראה כך:

```
<script type="text/javascript">
$(function () {
  $('.menuitems').hide();

  $('.menu').mouseover(function () {
    $(this).next().slideDown('slow').siblings('.menuitems').slideUp(200);
  });
});
</script>
```

השורה המרכזית כאן היא:

```
$(this).next().slideDown('slow').siblings('.menuitems').slideUp(200)
```

הסתייענו כאן בפונקציות לטיול על עץ ה-DOM ובשרשור אפקטים על-מנת לבצע את הפעולות הנדרשות. נפרק את השורה הזו לכמה חלקים על-מנת להבין את פעולתה:

1. בשלב ראשון אנחנו ניגשים לאלמנט העוקב בדף לאלמנט ה-`menu` הנוכחי (כלומר לתגית ה-`` העוקבת לתגית ה-`<p>` הנוכחית המתאימה לכותרת של התפריט שהמשתמש עומד עליו כרגע), וזאת באמצעות הקריאה לפונקציה:

```
$(this).next()
```

2. בשלב השני אנחנו מבצעים פתיחה של תגית ה-`` על-מנת להציג את הפריטים השייכים אליה, וזאת באמצעות הפונקציה `slideDown()`:

```
.slideDown('slow')
```

3. לאחר שאפקט ה- `slideDown` מסתיים, אנחנו מעוניינים להסתיר את כל התפריטים האחרים הפתוחים כרגע. לשם כך אנחנו נעזרים בפונקציה `siblings()` כדי לגשת לכל האלמנטים האחים לאלמנט ה- `menu` הנוכחי שמשויכים למחלקה `menuitems`.

```
.siblings('.menuitems')
```

4. לבסוף השתמשנו באפקט ה- `slideUp()` על מנת לסגור את כל התפריטים האחרים. הגדרנו קצב של 200 מילישניות עבור האנימציה.

```
.slideUp(200)
```

להלן התוצאה המתקבלת בדפדפן:



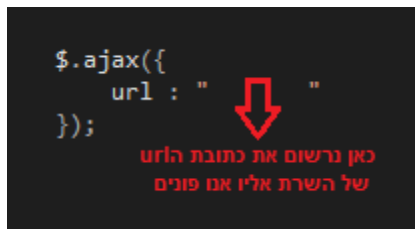
בדוגמא זו ראינו כיצד באמצעות מספר מצומצם של שורות קוד ב- `jQuery` הצלחנו להשיג אפקט די מרשים!

AJAX - ראשי תיבות של Asynchronous JavaScript and XML

טכנולוגית AJAX עוזרת לנו לטעון נתונים מהשרת ללא רענון דף הדפדפן. ו-JQuery מספקת קבוצה רחבה של פונקציות עזר לפיתוח קל ומהיר של בקשות AJAX. להלן האפשרויות הנפוצות:

12.1. פונקציית \$ajax

הפונקציה \$ajax יוצרת בקשת HTTP. תחביר בסיסי לשימוש בפונקציה זו:



נוכל להוסיף לאובייקט הנשלח לפונקציית \$.ajax, ערכים נוספים (אופציונליים) הנכתבים בצורת key/value:

שם ה-key	הערך שנכניס בתור value
async	מקבל ערך בוליאני המציין אם לבצע את הבקשה באופן אסינכרוני. ערך ברירת המחדל הוא true
timeout	מקבל מספר המייצג את הזמן המקסימלי (באלפיות שנייה) שיוקצב עבור הבקשה
type	מקבל מחרוזת המגדירה את שיטת ה-HTTP המשמשת עבור הבקשה (GET או POST) ערך ברירת המחדל הוא GET
contentType	מקבל מחרוזת המכילה סוג תוכן MIME להגדרת הבקשה. ערך ברירת המחדל הוא x-www-form-urlencoded
data	מקבל אובייקט json או מחרוזת שנשלח לשרת עם הבקשה.
dataType	מקבל מחרוזת המגדירה את סוג הנתונים שיוחזר מהשרת, XML, json, html וכו')
beforeSend	מקבל פונקציית callback שתבוצע לפני שליחת הבקשה
complete	מקבל פונקציית callback שתבוצע בסיום הבקשה
error	מקבל פונקציית callback שתבוצע אם הבקשה נכשלה
success	מקבל פונקציית callback שתבוצע אם הבקשה תצליח

לצורך הדוגמא, ניצור דף ASP.NET בשם default.aspx שיקבל שני מספרים דרך ה- QueryString ויחזיר את הסכום שלהם:

```
protected void Page_Load(object sender, EventArgs e)
{
    if (Request.QueryString["num1"] == null || Request.QueryString["num2"] == null)
        return;

    int num1 = int.Parse(Request.QueryString["num1"]);
    int num2 = int.Parse(Request.QueryString["num2"]);

    int sum = num1 + num2;
    Response.Write(sum);
    Response.End();
}
```

עתה ניצור דף HTML בשם default2.htm שבו באמצעות jQuery נבצע קריאה ב- GET לדף default.aspx ונציג על-ידי הודעת alert את תוצאת החישוב. לשם כך, נשתמש בפונקציה **ajax()** המקבלת את הפרמטרים הבאים:

```
$.ajax({
    type: "GET",
    url: "Default.aspx",
    data: "num1=3&num2=5",
    success: function (result) {
        alert(result);
    },
    error: function (error) {
        alert(error);
    }
});
```

נפתח את הדף default2.htm בדפדפן, והתוצאה היא:



12.2. פונקציית \$ajaxSetup

הפונקציה `ajaxSetup` קובעת הגדרות גלובליות לבקשות AJAX עתידיות. האופציות מורכבות מ- `key/value`. (כל האופציות שאפשר לשלוח בפונקציית `ajax`, יכולות להשלח גם בפונקציה זו, אבל אופציית ה- `url` היא אופציונלית ולא חובה לשלוח אותה כמו בפונקציית `ajax`)

תחביר בסיסי לשימוש בפונקציה זו:

```
$.ajaxSetup({
    // כאן נכנסות האופציות שנרצה להגדיר
    // בצורה גלובלית עבור כל בקשות האjax
});
```

לדוגמא, נוכל ליצור את הבקשה שיצרנו בדוגמא הקודמת, בצורה הבאה:

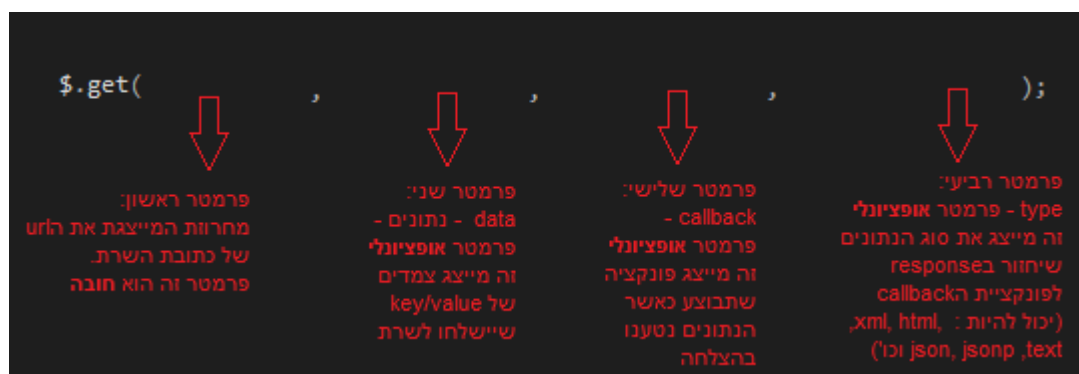
```
$.ajax({
    type: "GET",
    url: "Default.aspx"
});

$.ajax({
    data: "num1=3&num2=5",
    success: function (result) {
        alert(result);
    },
    error: function (error) {
        alert(error);
    }
});
```

get.12.3

הפונקציה get מקבלת נתונים מהשרת באמצעות בקשת http get request. (היא בעצם מהווה קיצור לבקשת get שנוכל לשלוח בפונקציית ajax)

תחביר בסיסי לשימוש בפונקציה זו:



מכיוון שהפונקציה מקבלת כפרמטר רק פונקציה אחת שתבצע כאשר הנתונים נטענו בהצלחה, קיימת תוספת לפונקציה המאפשרת לשרשר מיד בסיום הפונקציה `$.get` עוד אפשרויות לפונקציות callback שיתבצעו בהתאם למצב הבקשה:

```
$.get( " כתובת url ", function () {
  // פונקציה הנשלחת כפרמטר לפונקציית get ותבצע במקרה של הצלחה
})
.done(function () {
  // פונקציה שתבצע במקרה של הצלחה
})
.fail(function () {
  // פונקציה שתבצע במקרה של כישלון
})
.always(function () {
  // פונקציה שתבצע תמיד - גם במקרה של הצלחה וגם במקרה של כישלון
});
```

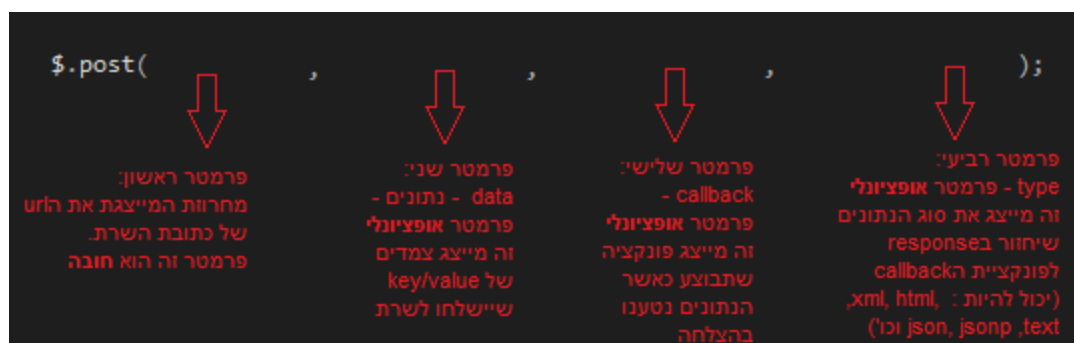
לדוגמא - נרשום את הבקשה שיצרנו בדוגמא הקודמת, בצורה הבאה:

```
$.get(
  "Default.aspx",
  "num1=3&num2=5",
  function (result) {
    alert(result);
  }
).fail(function (error) {
  alert(error);
});
```

12.4. post

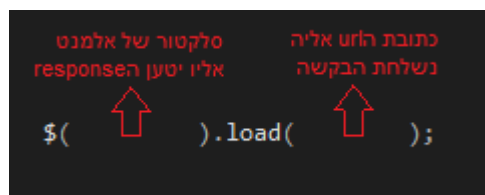
הפונקציה post מקבלת נתונים מהשרת באמצעות בקשת http post request.

תחביר בסיסי לשימוש בפונקציה זו:



12.5. load

תחביר בסיסי לשימוש בפונקציה זו:



12.6. JQuery AJAX Events

ניתן להגדיר אירועים שונים ב-JQuery ברמת הדף עצמו, שיבוצעו במהלך מחזור הפעולה של כל בקשת AJAX שניצור בדף.

האירועים מקושרים ל- ומתוזמנים עבור שלבים שונים בביצוע בקשת AJAX:

שם האירוע	תזמון האירוע	דוגמה למימוש האירוע
ajaxComplete	בכל פעם שבקשת AJAX תושלם	<pre>\$(document).ajaxComplete(function (event, xhr, settings) { alert("הושלם"); });</pre>
ajaxStart	בכל פעם שמתחילה בקשת AJAX	<pre>\$(document).ajaxStart(function () { alert("מתחיל"); });</pre>
ajaxError	בכל פעם שבקשת AJAX נכשלה	<pre>\$(document).ajaxError(function (event, jqxhr, settings, thrownError) { alert("נכשל"); });</pre>
ajaxSend	לפני שליחת בקשת AJAX	<pre>\$(document).ajaxSend(function (event, jqxhr, settings) { alert("מתחיל לשלוח"); });</pre>
ajaxStop	בכל פעם שבקשת AJAX הסתיימה	<pre>\$(document).ajaxStop(function () { alert("הסתיימה"); });</pre>
ajaxSuccess	בכל פעם שבקשת AJAX תושלם בהצלחה	<pre>\$(document).ajaxSuccess(function (event, request, settings){ alert("הצליח"); });</pre>



הערה: בתוך הפונקציה ששלחנו לארוע, נוכל להשתמש בפרמטרים שהפונקציה מקבלת, בכדי לגשת לנתוני פעולת ה-ajax הנוכחית שהפעילה את הארוע.

לדוגמא:

```
$(document).ajaxComplete(function (event, xhr, settings) {
    alert("the result is " + settings.url + ":" + xhr.responseText);
});
```

↓
↓

הזרז אליו הבקשה נשלחה
תוכן התשובה שחזרה

13. jQuery Plugins

בנוסף לפונקציונליות הקיימת בליבה של jQuery, ישנן ספריות נוספות של jQuery שניתן להוריד ולהשתמש בהן (ספריות אלה מכונות plugins). ניתן למצוא מאגרים גדולים של plugins בכתובות הבאות:

<http://plugins.jquery.com>

כל plugin ניתן להורדה כקובץ zip, ורבים מהם כוללים demos, קוד דוגמא והסברים כיצד להשתמש ב-plugin.

על-מנת להשתמש ב-plugin, בשלב ראשון יש לכלול את הספריות שלו בחלק ה-`<head>` של הדף, לאחר קובץ ה-jQuery הראשי:

```
<head>
  <title>Example</title>
  <script src="jquery.js" type="text/javascript"></script>
  <script src="jquery.plugin.js" type="text/javascript"></script>
</head>
```

ואז נוכל להשתמש בפונקציות המוגדרות בספריה של ה-plugin, כמו בכל קובץ JavaScript רגיל. בד"כ נוכל באמצעות הוספת שורה אחת בתוך הפונקציה של ה-`$(document).ready()` להפעיל את ה-plugin על אלמנט מסוים בדף, לדוגמא:

```
<script type="text/javascript">
```

```
$(document).ready(function () {
    $('#myID').somePlugin();
});
</script>
```

הרבה plugins מספקים פרמטרים אופציונליים נוספים המאפשרים לשנות את ההתנהגות של ה-plugin. בד"כ העברת הפרמטרים ל-plugin מתבצעת באמצעות אובייקט מפה של JavaScript (map) המורכב מזוגות של משתנים וערכים, המועבר כפרמטר לפונקציה של ה-plugin:

```
<script type="text/javascript">
    $(function () {
        $('#myID').somePlugin({
            send: false,
            message: 'This plugin is great!'
        });
    });
</script>
```

דוגמא לשימוש ב-plugin

להלן נראה דוגמא לשימוש ב-plugin המאפשר הצגת קרוסלה של תמונות מתחלפות. ה-plugin נקרא jquery feature carousel, וניתן להורדה מהכתובת:

<http://www.bkosborne.com/jquery-feature-carousel>

או ע"י המנח:

<https://www.npmjs.com/browse/keyword/jquery-plugin>

לאחר הורדת קובץ ה-zip של ה-plugin ופתיחת הקובץ, תקבלו את עץ הספריות הבא:

css	File Folder
images	File Folder
js	File Folder
CHANGELOG.txt	1 KB Text Document
GPL LICENSE.txt	35 KB Text Document
index.html	3 KB HTML Document
README.txt	1 KB Text Document

בתיקיית ה-js וה-css נמצאים כל קבצי ה-JavaScript וה-CSS הדרושים לצורך הפעלת ה-plugin. בתיקיית ה-images עליכם להכניס את כל התמונות שברצונכם להציג על הקרוסלה. התיקיה מכילה בהתחלה כמה תמונות לדוגמא.

הקובץ index.html מדגים את השימוש ב-plugin.

על-מנת להשתמש ב-plugin, עליכם ליצור תגית <div> עבור כל תמונה שברצונכם להציג על הקרוסלה, הכוללת בתוכה את הקישור לתמונה והטקסט שיופיע מתחת לתמונה, בפורמט הבא:

```
<div class="carousel-feature">
  
  <div class="carousel-caption">
    <p>Amazing flower</p>
  </div>
</div>
```

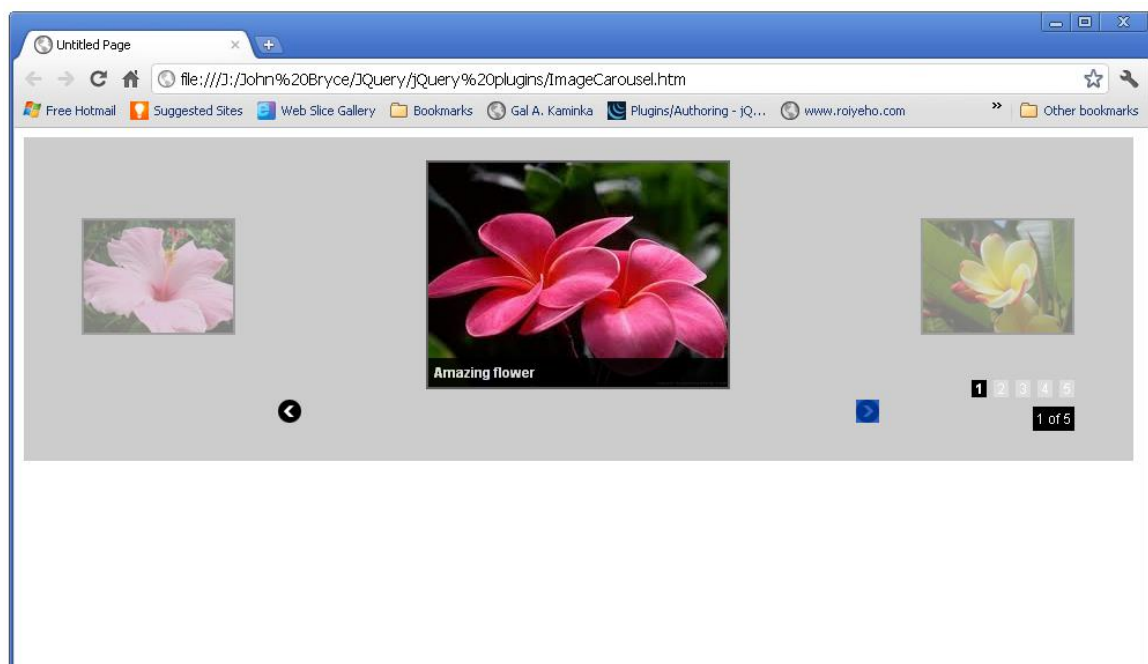
את כל ה-divים האלו עליכם לשים תחת תגית ה-<div> הראשית עם המזהה carousel. מבנה הדף המתקבל הוא:

```
<body>
  <div id="carousel-container">
    <div id="carousel">
      <div class="carousel-feature">
        
        <div class="carousel-caption">
          <p>Amazing flower</p>
        </div>
      </div>
      <div class="carousel-feature">
        
        <div class="carousel-caption">
          <p>Stunning flower</p>
        </div>
      </div>
      <div id="carousel-left"></div>
      <div id="carousel-right"></div>
    </div>
  </body>
```

לבסוף כדי להפעיל את ה-plugin, עליכם לקרוא לפונקציה featureCarousel() המוגדרת בקובץ :featureCarousel.js

```
<script src="jquery.featureCarousel.js" type="text/javascript"></script>
<script type="text/javascript">
  $(document).ready(function () {
    $("#carousel").featureCarousel();
  });
</script>
```

התוצאה המתקבלת בדפדפן היא:



14. jQuery UI

קיימת ספריה הנקראת jQuery UI, שפותחה על-ידי הצוות של jQuery, הכוללת מגוון גדול של פקדים כמו accordion, date picker, dialog, slider, tabs ומגוון רחב של אפקטים מתקדמים. ניתן להוריד את הספריה הזו בכתובת:

<http://ui.jquery.com>

הספריה עצמה שוקלת כ-500Kb, ולכן לא מומלץ להתקין את כולה בסביבת ה-production. בזמן ההורדה אפשר לבחור בהורדת custom, ולבחור רק את ה-features שבהם אתם מעוניינים להשתמש.

פקדים הקיימים בספריה:

- Accordion
- Autocomplete
- Button
- Checkboxradio
- Controlgroup
- Datepicker
- Dialog
- Menu
- Progressbar
- Selectmenu
- Slider
- Spinner
- Tabs
- Tooltip

אפקטים הקיימים בספריה:

- Add Class
- Color Animation
- Easing
- Effect
- Hide
- Remove Class
- Show
- Switch Class
- Toggle
- Toggle Class

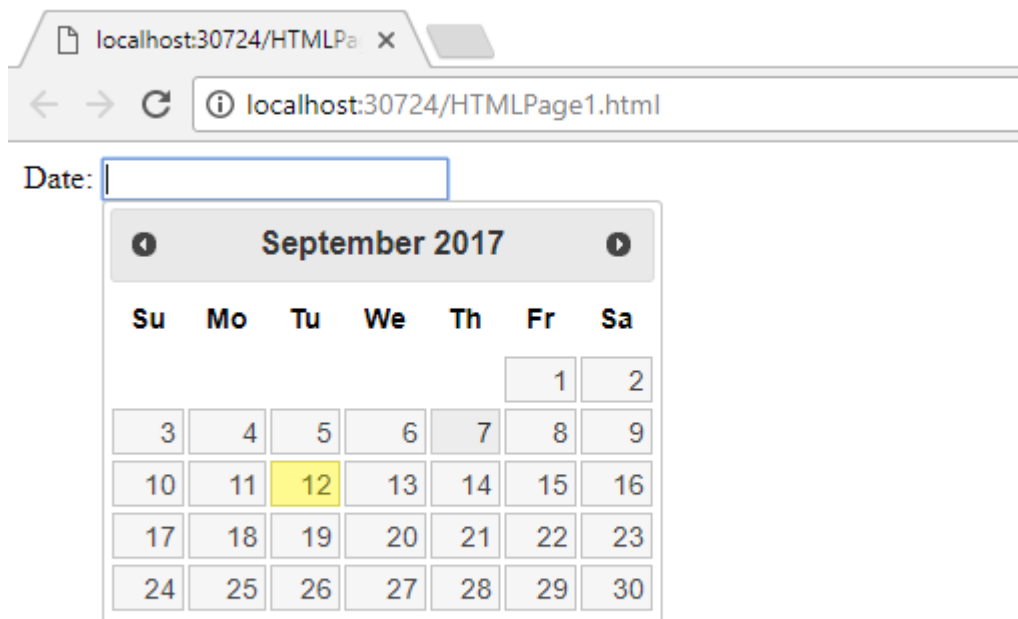
הוספת פונקציונליות לפקדים:

- Draggable
- Droppable
- Resizable
- Selectable
- Sortable

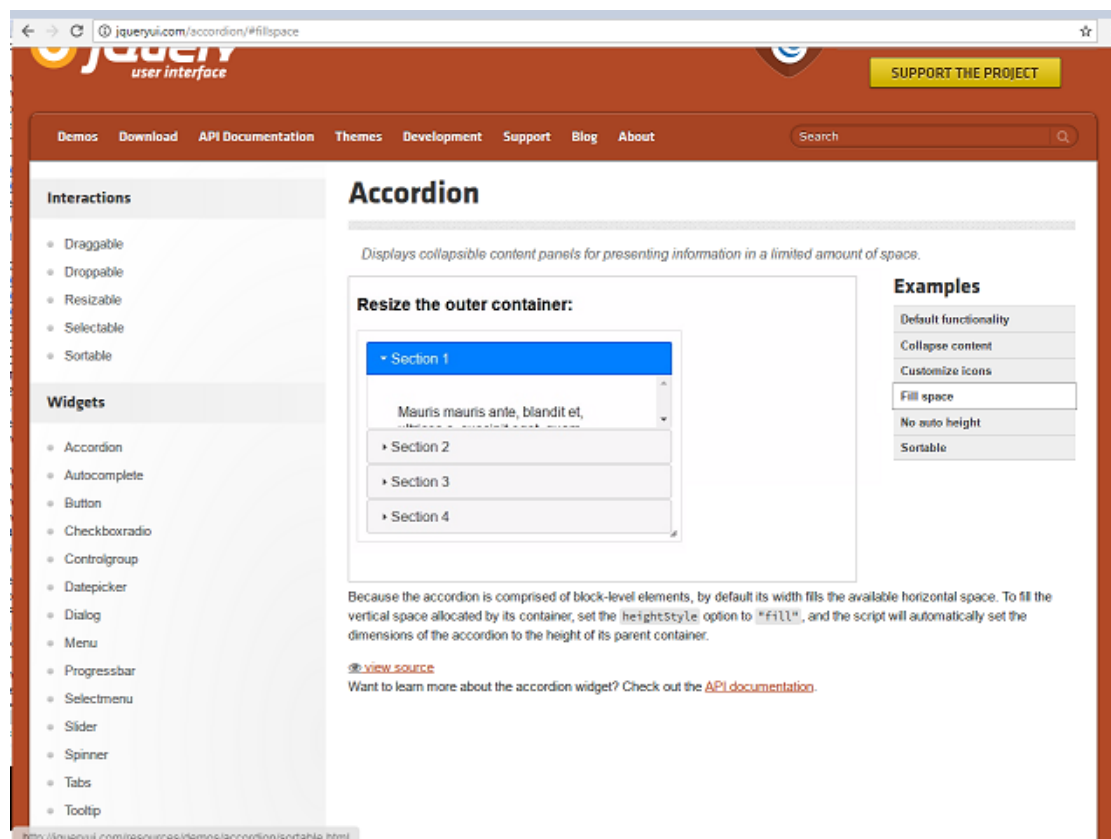
דוגמא לשימוש בפקד datePicker מתוך ספריית ה-jQuery UI:

```
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
  <title></title>
  <link rel="stylesheet" href="//code.jquery.com/ui/1.12.1/themes/base/jquery-ui.css">
  <script src="//ajax.googleapis.com/ajax/libs/jquery/3.2.1/jquery.min.js"></script>
  <script src="https://code.jquery.com/ui/1.12.1/jquery-ui.js"></script>
  <script type="text/javascript">
    $(function () {
      $('#datepicker').datepicker();
    });
  </script>
</head>
<body>
  <div>
    <p>Date: <input type="text" id="datepicker" /></p>
  </div>
</body>
</html>
```

להלן התוצאה המתקבלת בדפדפן:



באתר ה-JQuery UI עצמו קיימים הדגמות לשימושים שונים בפקד, וכן דוקומנטציה מפורטת עבור כל פונקציה ופרמטר שקיימים לפקד, לדוגמא שימוש ב-accordion:



15. כתיבת Custom Plugins

כתיבת plugins חדשים מאפשרת להרחיב את ספריית ה-JQuery ולהוסיף לה רכיבים ופונקציות חדשים, אשר ניתן לבצע בהם שימוש חוזר ועל-ידי-כך לחסוך כתיבה של קוד. את ה-plugin נכתוב בד"כ בתוך קובץ סקריפט חיצוני (עם סיומת js).

ברמה הבסיסית, כתיבה של jQuery plugin מתבצעת על-ידי הוספת פונקציה כמאפיין (property) חדש של האובייקט jQuery.fn. שם המאפיין יקבע את שם ה-plugin.

```
jQuery.fn.myPlugin = function () {
    // Write your plugin code here
}
```

אולם במקום לכתוב בצורה ישירה האובייקט jQuery יהיה לנו יותר נוח להשתמש בסימן ה-\$ המקובל. על-מנת למנוע התנגשות עם ספריות אחרות שעלולות גם כן להשתמש בסימן ה-\$ בזמן הפעלת ה-plugin, מומלץ להעביר את אובייקט ה-JQuery כפרמטר לפונקציה שתמפה אותו לסימן ה-\$, כך שסימן ה-\$ לא יוכל להידרס על-ידי ספריות אחרות בזמן ההרצה של ה-plugin. מכאן שכתובת ה-plugin תתבצע באופן הבא:

```
(function ($) {
    $.fn.myPlugin = function () {

        // Write your plugin code here

    };
})(jQuery);
```

הפונקציה העוטפת נכתבת בתוך סוגריים, כיוון שמיד לאחר כתיבתה אנחנו מבצעים הפעלה שלה ומעבירים לה את אובייקט ה-JQuery כפרמטר המתמפה לסימן ה-\$.

עתה בתוך הפונקציה הפנימית נוכל להשתמש בסימן ה-\$ במקום אובייקט ה-JQuery ככל שנרצה. יתרון נוסף לצורת כתיבה זו הוא שפונקציות פנימיות שנשתמש בהן לצורך מימוש ה-plugin לא יהיו חשופות כלפי חוץ, ולכן לא יעמיסו על מרחב השמות הכללי.

נשים לב שבתוך הפונקציה הפנימית המילה this מתייחסת לאובייקט ה-JQuery שעליו הופעל ה-plugin, ולא לאלמנט ה-DOM הפנימי. לכן כדי להתייחס לאובייקט שעליו אנחנו רוצים לבצע את הפעולה בתוך הפונקציה, אין צורך לעטוף את המילה this בתוך סימן ה-\$ (כלומר לרשום \$(this)), זאת בניגוד למקרים אחרים של רישום פונקציות callback לאירועים של jQuery (כמו אירוע ה-click).

נראה דוגמא ל-plugin פשוט בשם maxHeight, אשר יחזיר את הגובה של האלמנט הגבוה ביותר בקבוצת האלמנטים הנוכחית:

```
(function ($) {
    $.fn.maxHeight = function () {

        var max = 0;
        this.each(function () {
            var currHeight = $(this).height();
            if (currHeight > max)
                max = currHeight;
        });

        return max;
    };
})(jQuery);
```

עתה נוכל להפעיל את ה-plugin שרשמנו על כל אובייקט של jQuery, לדוגמא נוכל להפעיל אותו על קבוצת כל אלמנטי ה-`<div>` השייכים למחלקה `column` באופן הבא:

```
var tallest = $('div.column').maxHeight();
```

כדי לבדוק את ה-plugin, נבנה את הדף הבא שיכיל 3 עמודות בעלות גבהים שונים (בהתאם לכמות התוכן בכל עמודה), ויפעיל את הפונקציה `maxHeight()` לאחר טעינת הדף:

```
<html xmlns="http://www.w3.org/1999/xhtml" >
<head>
  <title>jQuery Custom Plugin</title>
  <style type="text/css">
    #wrapper
    {
      width: 760px;
      margin: 0 auto;
    }
    .column
    {
      float: left;
      padding: 10px;
    }
    #col1
    {
      width: 110px;
      margin-right: 10px;
      background-color: #E2DDC4;
    }
    #col2
    {
      width: 200px;
      margin-right: 10px;
      background-color: #B5C9DA;
    }
    #col3
    {
      width: 210px;
      background-color: #E87C5E;
    }
  </style>

  <script type="text/javascript" src="jquery-3.2.1.js"></script>
  <script type="text/javascript" src="jQuery.maxHeight.js"></script>
  <script type="text/javascript">
    $(function () {
      var tallest = $('div.column').maxHeight();
```

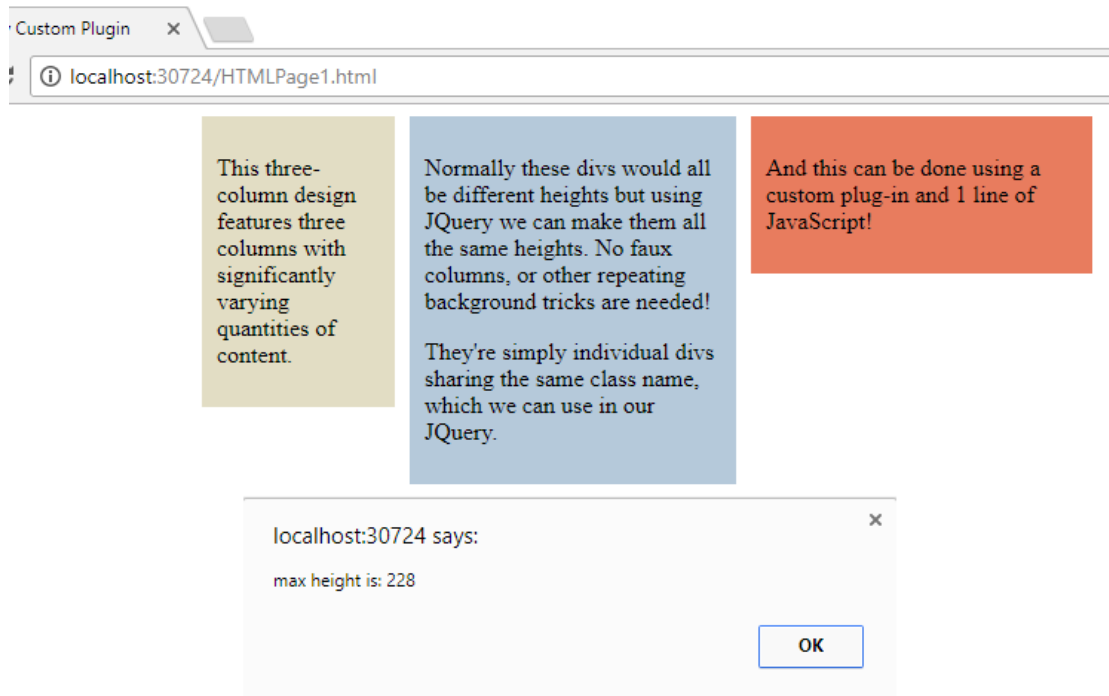


```

        alert('max height is: ' + tallest);
    });
</script>
</head>
<body>
    <div id="wrapper">
        <div class="column" id="col1">
            <p>This three-column design features three columns with
significantly varying quantities of content.</p>
        </div>
        <div class="column" id="col2">
            <p>Normally these divs would all be different heights but using
JQuery we can make them all the same heights. No faux columns, or other
repeating background tricks are needed!</p>
            <p>They're simply individual divs sharing the same class name,
which we can use in our JQuery.</p>
        </div>
        <div class="column" id="col3">
            <p>And this can be done using a custom plug-in and 1 line of
JavaScript!</p>
        </div>
    </div>
</body>
</html>

```

התוצאה המתקבלת בדפדפן היא:



במקרה זה ה-plugin החזיר מספר שלם בתור התוצאה, אולם בהרבה מקרים נרצה שה-plugin יבצע שינוי כלשהו על אוסף האלמנטים, ויעביר אותם הלאה לפונקציה הבאה המופעלת בשרשרת הקריאות. על-מנת לתמוך בהפעלת ה-plugin כחלק משרשרת קריאות (ראה סעיף "שרשור אירועים"), עלינו לדאוג לכך שה-plugin יחזיר את האובייקט `this` בסוף הפונקציה.

לדוגמא, נשנה את ה-plugin שכתבנו, כך שבמקום להחזיר את הגובה של העמודה הכי גבוהה, ישנה את הגובה של כל העמודות כך שהוא יהיה זהה לגובה של העמודה הגבוהה ביותר (נקרא ל-plugin החדש בשם `equalHeight`):

```
(function ($) {
    $.fn.equalHeight = function () {

        var max = 0;
        this.each(function () {
            var currHeight = $(this).height();
            if (currHeight > max)
                max = currHeight;
        });

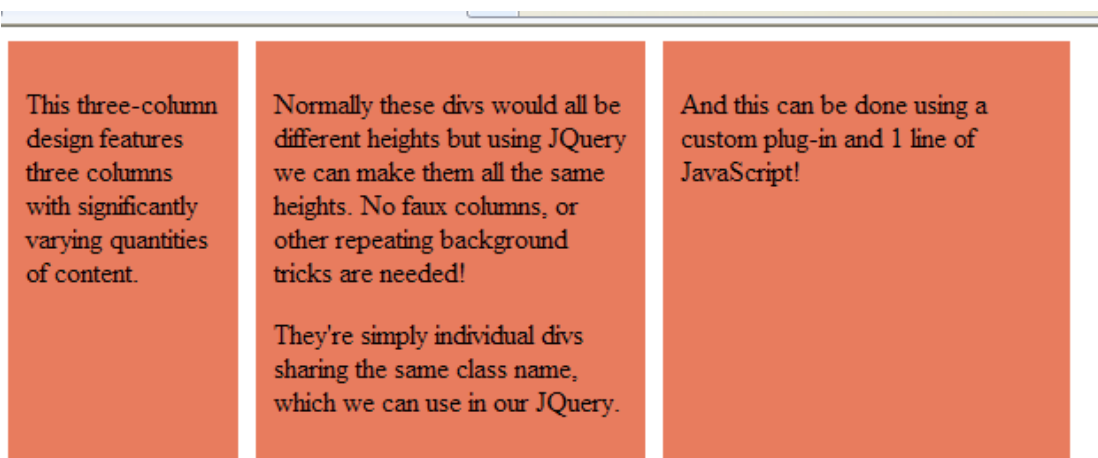
        this.height(max);
        return this;
    };
})(jQuery);
```

עתה נוכל להפעיל את ה-plugin כחלק משרשרת קריאות על אובייקט ה-jQuery, למשל:

```
$('#div.column').equalHeight().css('background-color', '#E87C5E');
```

בזכות העובדה שה-plugin מחזיר את אובייקט ה-`this`, נוכל להמשיך להפעיל על אוסף האובייקטים פונקציות נוספות של jQuery כמו `css`.

התוצאה המתקבלת בדפדפן היא:



ניתן לקבל מידע נוסף על כתיבת plug-ins והרחבות ל- jQuery בכתובת הבאה:

<http://docs.jquery.com/Plugins/Authoring>



16. תרגילים

תרגיל 1



צרו דף עם רשימה של 5 קישורים.

בזמן טעינת הדף הוסיפו באמצעות jQuery בצורה דינמית לדף 5 לחצנים – לחצן אחד ליד כל קישור. כל לחצן יציג את הכתובת שאליו מוביל הקישור.

תרגיל 2



צרו דף עם גלריית תמונות (שתכיל לפחות 4-5 תמונות).

יש להגדיר גודל תמונה זהה בעזרת class מתאים (גובה ורוחב).

הוסיפו באמצעות jQuery אפקט לדף כך שכאשר המשתמש עובר עם העכבר מעל הגלריה, כל תמונה תוחלף בתמונה שבאה אחריה בגלריה (לדוגמא אם יש בגלריה 4 תמונות – תמונה מס' 1 תהפוך להיות תמונה מס' 2, תמונה מס' 2 תהפוך להיות תמונה מס' 3, תמונה מס' 3 תהפוך להיות תמונה מס' 4 ותמונה מס' 4 תהפוך להיות תמונה מס' 1).

תרגיל 3



צרו דף עם הטבלה הבאה:

Superfoods

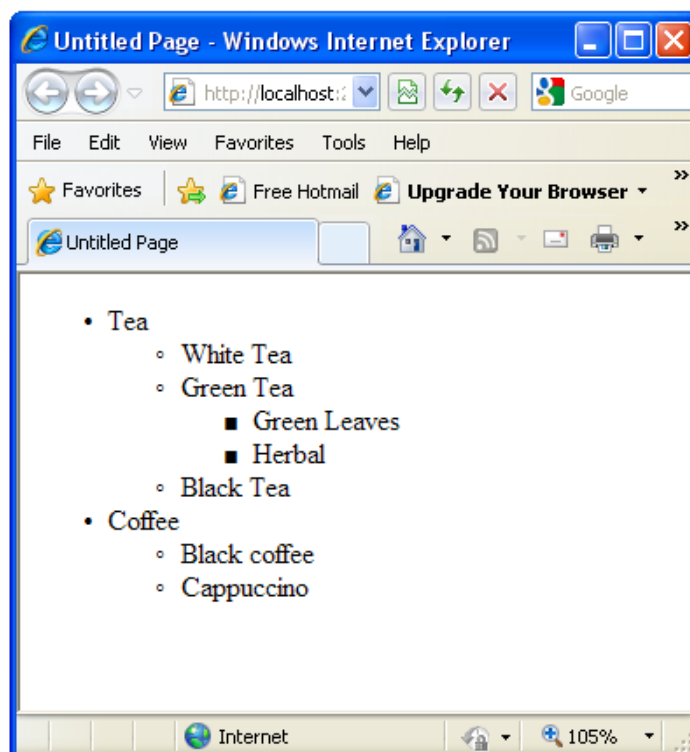
Salmon	Omega-3's help the brain develop properly, reduce the risk of Alzheimer's, and help prevent heart disease.
Spinach	Great source of folate and lutein. Prevents birth defects, heart disease, stroke, and protects your skin from sun damage.
Sweet Potatoes	Beta carotene protects your skin from sun damage.
Beans	Packed with protein, fiber and antioxidants, beans contribute to weight management and lower risk for heart disease and cancer.
Berries	Antioxidants and phytochemicals prevent cancer, Alzheimer's and heart disease.
Green Tea	Antioxidants lower risk for heart disease, cancer, and stroke.

הגדירו צבע רקע שונה לכל השורות האי-זוגיות בטבלה. את הצביעה הראשונית יש לבצע באמצעות jQuery בזמן טעינת הדף.
הוסיפו לחצן בדף, כך שבכל לחיצה בדף צביעת השורות בטבלה תשתנה מצביעה של השורות האי-זוגיות לצביעת השורות הזוגיות ולהיפך.

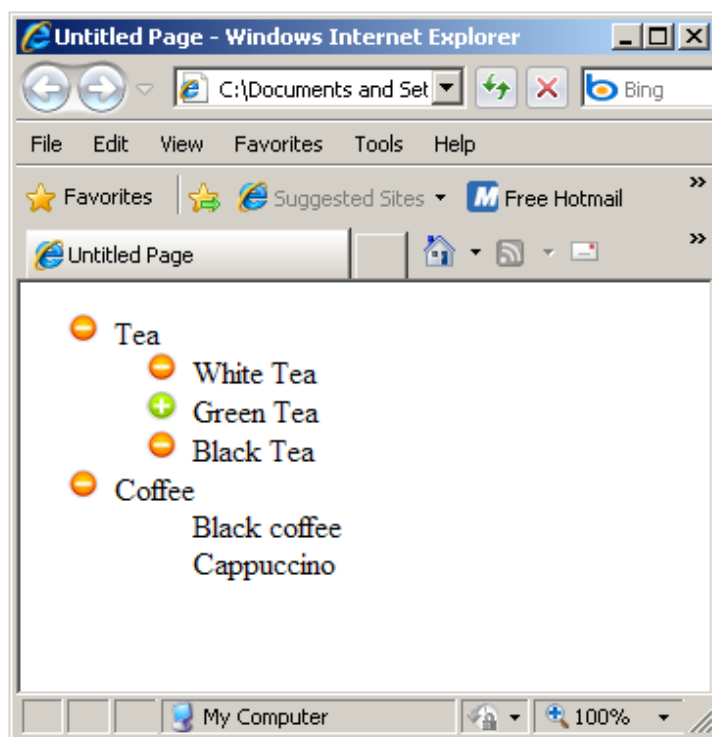
תרגיל 4



נתון דף המכיל רשימות מקוננות של פריטים.
ליד כל פריט המכיל תתי-פריטים עליכם להוסיף אייקון של + מצד שמאל. כאשר לוחצים על האייקון, עליכם להרחיב את רשימת תתי-הפריטים שלו, ולהחליף את סימן ה- + בסימן -.
כאשר לוחצים על אייקון - עליכם לסגור את רשימת תתי-הפריטים.
רמז: השתמשו בתכונת ה- `list-style-image` CSS כדי לקבוע את האייקון.



כך אמור להיראות הדף אחרי הפעלת קוד ה-jQuery:



תרגיל 5

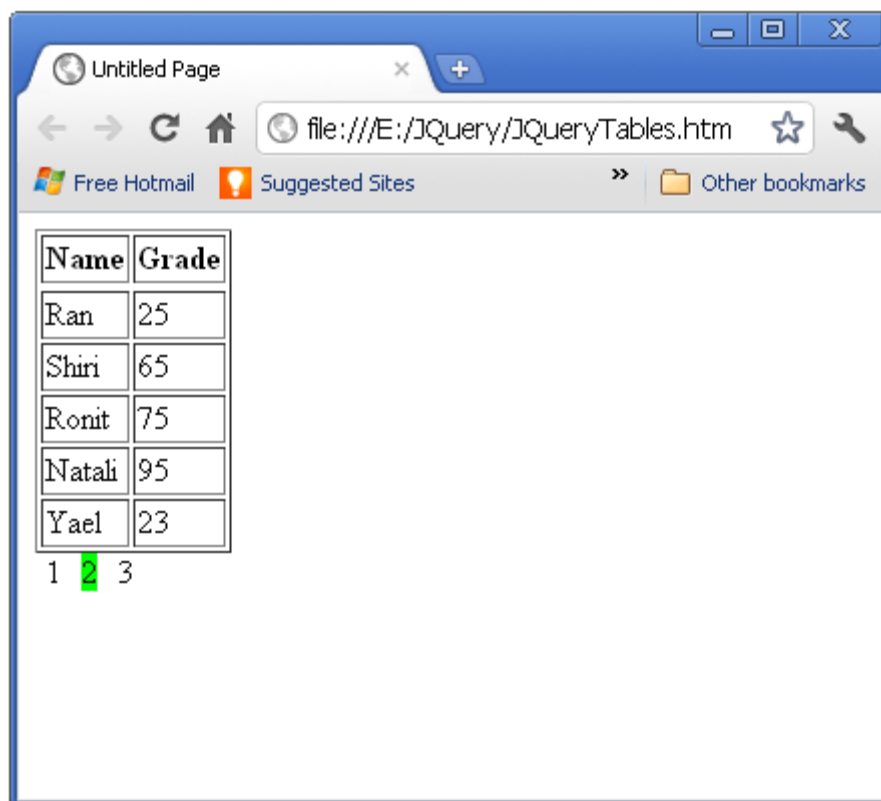


בנו דף שיציג רשימת נכסים למכירה.
עבור כל נכס יוצגו הפרטים הבאים: מיקום, מספר חדרים, שווי הנכס, פרטי איש קשר.
יש להציג את הנכסים באמצעות פקד ה-accordion. השתמשו לשם כך בספריית jQuery UI.

תרגיל 6 (למתקדמים)



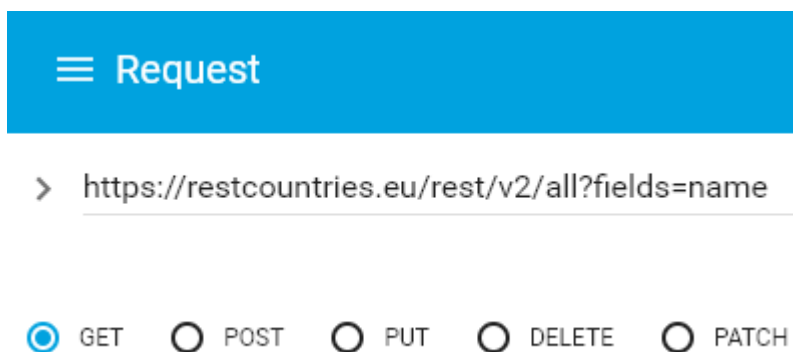
נתון דף המציג טבלה עם נתונים.
באמצעות jquery עליכם להוסיף יכולת paging לטבלה, כלומר לחלק את הטבלה לדפים, כך שכל אחד מהם יציג X רשומות מתוך הטבלה (למשל 5 רשומות בכל דף). בתחתית הטבלה יופיעו לינקים שיאפשרו למשתמש לעבור בין הדפים.
לדוגמא דף המציג רשימת ציונים של סטודנטים יראה כך:



תרגיל 7 (למתקדמים)



נתונה הבקשה הבאה:



הבקשה נשלחת באמצעות get, לכתובת הנתונה למעלה. ומחזירה מערך של אובייקטי json בתבנית הבאה:

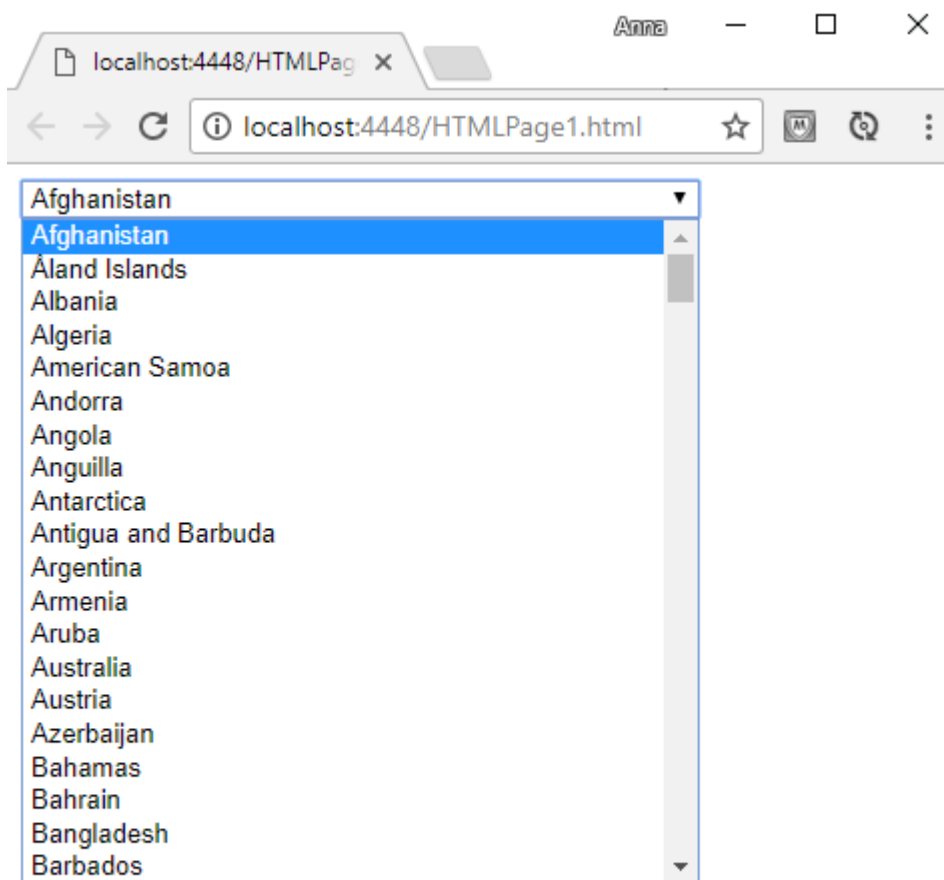
```
{"name": " " }
```

להלן חלק מהתשובה שחזרה:

```
[
  { "name": "Afghanistan" },
  { "name": "Åland Islands" },
  { "name": "Albania" },
  { "name": "Algeria" },
  { "name": "American Samoa" },
  { "name": "Andorra" },
  { "name": "Angola" },
  { "name": "Anguilla" },
  { "name": "Antarctica" },
  { "name": "Antigua and Barbuda" },
  { "name": "Argentina" },
  { "name": "Armenia" },
  { "name": "Aruba" },
  { "name": "Australia" },
  { "name": "Austria" },
  { "name": "Azerbaijan" },
  { "name": "Bahamas" },
  { "name": "Bahrain" },
  { "name": "Bangladesh" },
  { "name": "Barbados" },
  { "name": "Belarus" },
  { "name": "Belgium" },
  { "name": "Belize" },
  { "name": "Benin" },
  { "name": "Bermuda" },
  { "name": "Bhutan" },
  { "name": "Bolivia (Plurinational State of)" },
  { "name": "Bonaire, Sint Eustatius and Saba" },
  { "name": "Bosnia and Herzegovina" },
  { "name": "Botswana" },
  { "name": "Bouvet Island" },
  { "name": "Brazil" },
  { "name": "British Indian Ocean Territory" },
  { "name": "United States Minor Outlying Islands" },
  { "name": "Virgin Islands (British)" },
```

עליכם לבצע פניית ajax באמצעות jquery, ואת המערך שהתקבל לתת בתור אופציות בחירה עבור פקד options.

כלומר, צרו בדף תתקבל התוצאה הבאה:



תרגיל 8 (למתקדמים)

המשך לתרגיל 7: שכללו את אופצית בחירת המדינה מהנתונים שחזרו ע"י בקשת ה- ajax, לתיבת קלט Autocomplete, ע"י שימוש בפלאגין הבא של ספריית ui – jquery (הלינק לפלאגין הוא



:(<https://jqueryui.com/autocomplete>)

Autocomplete

Enables users to quickly find and select from a pre-populated list of values as they type, leveraging searching and filtering.

Tags:

- BASIC
- COBOL
- Ruby

Examples

Default functionality

Accent folding

Categories

Combobox

Custom data and display

Multiple values

Multiple, remote

Remote datasource

Remote JSONP datasource

Remote with caching

Scrollable results

17. פתרונות לתרגילים נבחרים

פתרון תרגיל 4

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
  <title>Untitled Page</title>
  <script src="jquery-1.7.min.js" type="text/javascript"></script>
  <style type="text/css">
    .plusimageapply
    {
      list-style-image: url(images/plus.png);
    }
    .minusimageapply
    {
      list-style-image: url(images/minus.png);
    }
    .noimage
    {
      list-style-type: none;
      list-style-image: none;
    }
  </style>
  <script type="text/javascript">
    $(document).ready(function () {
      $('li').each(function () {
        var $this = $(this);
        if ($this.children().size() == 0) {
          $this.addClass('noimage');
          return;
        }

        $this.addClass('plusimageapply');
        $this.children().hide();

        $this.click(function (event) {
          var $this = $(this);
          if ($this.is('.plusimageapply')) {
            $this.children().show();
            $this.removeClass('plusimageapply');
            $this.addClass('minusimageapply');
          }
          else {
            $this.children().hide();
            $this.removeClass('minusimageapply');
            $this.addClass('plusimageapply');
          }
          event.stopPropagation();
        });
      });
    });
```

```
});  
</script>  
</head>  
<body>  
  <ul>  
    <li>Tea  
      <ul>  
        <li>White Tea</li>  
        <li>Green Tea  
          <ul>  
            <li>Green Leaves</li>  
            <li>Herbal</li>  
          </ul>  
        </li>  
        <li>Black Tea</li>  
      </ul>  
    </li>  
    <li>Coffee  
      <ul>  
        <li>Black coffee</li>  
        <li>Cappuccino</li>  
      </ul>  
    </li>  
  </ul>  
</body>  
</html>
```

פתרון תרגיל 6

```

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" >
<head>
  <title>Untitled Page</title>
  <style type="text/css">
    .highlight
    {
      background-color: Lime;
    }
    .page
    {
      margin: 5px;
    }
  </style>

  <script src="jquery-1.4.2.js" type="text/javascript"></script>
  <script type="text/javascript">
    function createPages() {
      var recPerPage = 5;
      var recNum = $('tbody tr').length;
      var pagesNum = Math.ceil(recNum / recPerPage);

      $('<div id="pages"></div>').insertAfter('table');
      for (var i = 1; i <= pagesNum; i++) {
        $('<span class="page">' + i + '</span>').appendTo($('#pages')).click(
          function () {
            $('tbody tr').hide();
            var currPage = $(this).text();

            for (var j = (currPage - 1) * recPerPage; j <= currPage *
recPerPage - 1; j++) {
              //alert(j);
              $('tbody tr').eq(j).show();
            }
          });
      }

      $('#pages span').hover(function () {
        $(this).addClass('highlight');
      },
        function () {
          $(this).removeClass('highlight');
        });

      $('#pages span').eq(0).click();
    }

    $(document).ready(function () {
      enableColumnSorting();
    });
  </script>
</head>
<body>
  <table border="1" class='myTable'>

```

```

<thead>
  <tr><th>Name</th><th>Grade</th></tr>
</thead>
<tbody>
  <tr><td>Alon</td><td>85</td></tr>
  <tr><td>Avi</td><td>65</td></tr>
  <tr><td>Moshe</td><td>92</td></tr>
  <tr><td>Gal</td><td>44</td></tr>
  <tr><td>David</td><td>24</td></tr>
  <tr><td>Ran</td><td>25</td></tr>
  <tr><td>Shiri</td><td>65</td></tr>
  <tr><td>Ronit</td><td>75</td></tr>
  <tr><td>Natali</td><td>95</td></tr>
  <tr><td>Yael</td><td>23</td></tr>
  <tr><td>Hadas</td><td>44</td></tr>
  <tr><td>Gili</td><td>46</td></tr>
</tbody>
</table>
</body>
</html>

```

פתרון תרגיל 7

```

<!DOCTYPE html>

<html lang="en" xmlns="http://www.w3.org/1999/xhtml">
<head>
  <meta charset="utf-8" />
  <title></title>

  <script src="//ajax.googleapis.com/ajax/libs/jquery/3.2.1/jquery.min.js"></script>

  <script>
    $(function () {
      $.ajax({
        url: 'https://restcountries.eu/rest/v2/all?fields=name',
        type: 'GET',
        dataType: 'json',
        success: function (json) {
          $.each(json, function (i, value) {
            $('#myselect').append($('').text(value.name).attr('value', value.name));
          });
        }
      });
    });
  </script>
</head>
<body>
  <select id="myselect"></select>
</body>
</html>

```

18. מקורות לקריאה נוספת

jQuery website

<http://jquery.com>

jQuery wiki

<http://docs.jquery.com>

jQuery API

<http://remysharp.com/jquery-api>

The jQuery blog

<http://jquery.com/blog>

Learning jQuery

<http://www.learningjquery.com>

JavaScript Toolbox

<http://www.javascripttoolbox.com>

MSDN Jscript Reference

[http://msdn.microsoft.com/en-us/library/x85xxsf4\(VS.71\).aspx](http://msdn.microsoft.com/en-us/library/x85xxsf4(VS.71).aspx)

