שיעור 8

מתחילים...

Object

אובייקטים – סוג של מבנה נתונים אשר מחזיק כמה מאפיינים ובכך מאפשר שמירה של נתונים תחת ארגומנט אחד.

```
let student = {
firstName: ‘dor’,
lastName:
‘dekel’,
ID:
‘123456789’,
GPA: 82.7
};
```

Object

גישה לכל מאפיין מתבצעת באופן הבא:

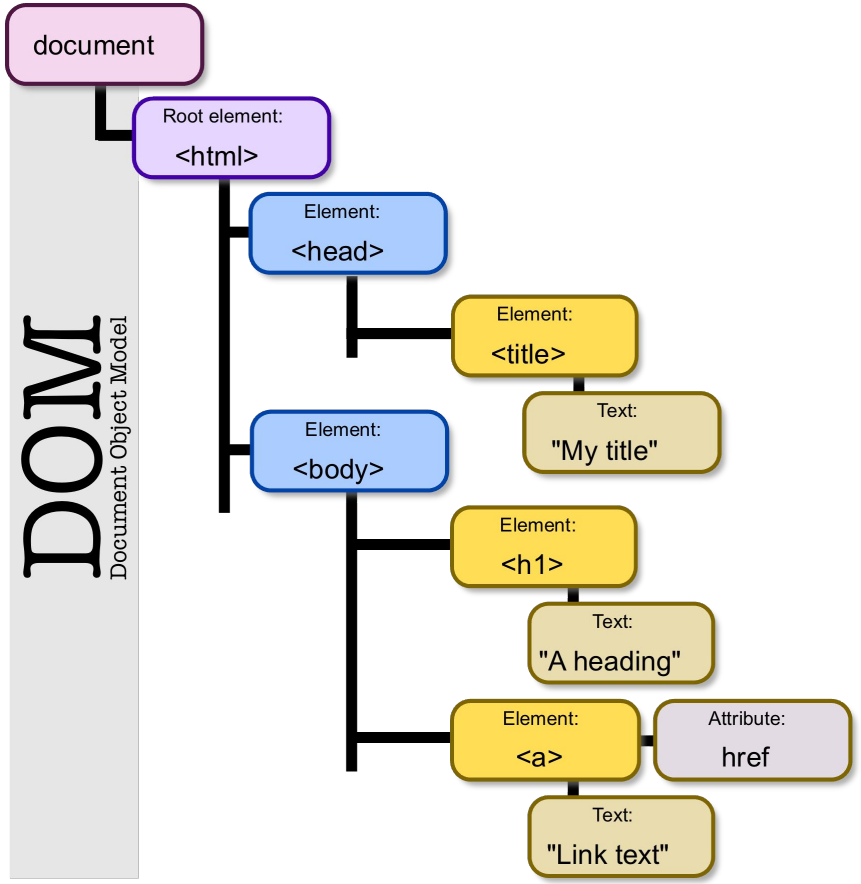
nameOfObject.property

(בהמשך לדוגמא הקודמת):

```
student.lastName = ‘levi’;
document.getElementById(‘..’).innerHTML =
student.lastName;
```

DOM

Document object model



באמצעות DOM ל JSיש את היכולת לגשת ולשנות את כל האלמנטים של מסמך ה HTML.

הדפדפן יוצר דיאגרמה של עץ המציג אובייקטים יחד עם הקשרים והמידע של כל אלמנט בדף HTML.

ובכך מקבל הJavaScript את הכוח ליצירת דף HTML דינמי.

Document Methods

Find Element

פעולה	הסבר
document.getElementById(id)	מוצא אלמנט לפני id
document.getElementsByTagName(name)	מוצא אלמנט לפני תגית
document.getElementsByClass (name)	מוצא אלמנט לפני class

Document Property

Edit Element

פעולה	הסבר
element.innerHTML = new html content	משנה את התוכן של האלמנט
element.attribute = new value	משנה את ערך התכונה של האלמנט
element.setAttribute(attribute, value)	
element.style.property = new style	משנה את העיצוב של האלמנט

Document Methods

Adding and Deleting Elements

פעולה	הסבר
document.createElement(element)	צור אלמנט חדש
document.removeChild(element)	הסר אלמנט
document.appendChild(element)	צרף אלמנט
document.replaceChild(new, old)	החלף אלמנט ישן באלמנט חדש

Class

מחלקות נוספו לשפה כאשר יצא העדכון של ES6, מחלקות ב JS הן תבניות לאובייקטים של JS, כלומר כאשר יש מחלקה ניתן ליצור אובייקט בעזרתו, להשתמש במאפייניו ופעולותיו.

בכדי ליצור אובייקט עם כל המאפיינים, נשתמש בבנאי constructor שיכריח להזין את הערכים לכל המאפיינים.

הבנאי נקרא אוטומטית כאשר נוצר אובייקט חדש.

Class Constructor & Properties

בדוגמה זו יצרנו מחלקת מוצר ובנאי שמקבל 2 ערכים: שם מוצר ומחיר המוצר.

```
class Product {
  constructor(name, price) {
    this.name = name;
    this.price  = price;
  }
}
```

Class Methods


```
class Product {
  constructor...

  discount() {
    return this.price * 0.9;
  }
}
```

נוסיף למחלקה גם מתודה לחישוב הנחה, כך שכל מוצר יקבל את אותה המתודה ויציג את המחיר לאחר ההנחה, בהתאם למחיר של המוצר. הרי המחירים של המוצרים שונים וכך גם ההנחה.

Object Instance

בעת יצירת מוצר חדש העברנו לבנאי את הערכים המתאימים.
כעת יש לנו 2 מוצרים: במבה וביסלי עם המחירים 4 ו5 בהתאמה.

```
let p1 = new Product(”Bamba", 5);
let p2 = new Product(”Bisli", 4);
```

```
console.log(`Product name: ${p1.name}, the price after discount is ${p1.discount()}`)
console.log(`Product name: ${p2.name}, the price after discount is ${p2.discount()}`)
```

Conso
le

```
Product name: Bamba, the price after discount is 4.5
Product name: Bisli, the price after discount is 3.6
```

Inheritance

בעת הורשה אנו "מורישים" (מעבירים) את כל המאפיינים והמתודות למחלקה היורשת בעת יצירת אותה מחלקה חדשה, הורשה שימושית עבור שחזור קוד.

לשם יצירת הורשה נכיר את המילה השמורה extends אשר בעזרתה נגדיר את המחלקה היורשת ממחלקת הבסיס (המורישה).

Inherits Constructor

```
class Category {
  constructor(categoryName) {
    this.categoryName  =
categoryName;
  }
  getCategoryName() {
    return this.categoryName;
  }
}
```

```
class Product extends Category {
   constructor(name, price,
categoryName) {
      super(categoryName)
      this.name = name;
      this.price  = price;
  }
  discount() {
     return this.price * 0.9;
  }
}
```

פקודת super() מעבירה את הערך אל הבנאי של מחלקת הבסיס
כעת כאשר נשתמש בבנאי Product בעת יצירת אובייקט נצטרך להעביר 3 ערכים.

Use Inherits Methods

לאחר שירשנו את המחלקה Category אתה ירשנו גם את המתודות והמאפיינים.

מסיבה זו נוכל להשתמש במתודה הקיימת במחלקת הבסיס לאחר שניצור אובייקט של Product. הנה דוגמה:

```
let p1 = new Product("Bamba", 5, "Snacks")
let p2 = new Product("Yogurt", 4, "Dairy Products")

console.log(`${p1.name}, belongs to the category ${p1.getCategoryName()}`)
console.log(`${p2.name}, belongs to the category ${p2.getCategoryName()}`)
```

```
Bamba, belongs to the category Snacks
Yogurt, belongs to the category Dairy Products
```

Localstorage / Sessionstorage

- javascript שמירת מידע בדפדפן באמצעות יכולות אחסון של .
- **הוא בכך ששימוש ב, sessionStorage ו localStorage ההבדל בין אינו שומר את הנתונים כאשר המשתמש סוגר את sessionStorage הדפדפן.**

על מנת שנוכל לשמור את הנתונים עלינו לבצע 2 פעולות :

1. **הגדרה ושמירת המידע.**
2. **קריאה וקבלת המידע שנשמר .**

Localstorage

```
50    let student = 'yardy almog';
51
52    //  user  רכובר המשתמש לתוך
53    localStorage.setItem('user', student);
54
```

```
1
2    //
3    let getStudent = localStorage.getItem('user');
4
5    console.log(getStudent);
6
```

Sessionstorage

```
49
50  let student = 'yaniv almog';
51
52  //  user  שומר את המשתנה student
53  sessionStorage.setItem('user', student);
54
```

```
1
2  //  שליפת המשתנה student מהזיכרון
3  let getStudent = sessionStorage.getItem('user');
4
5  console.log(getStudent);
6
```

סוף שיעור 8

האם יש שאלות?