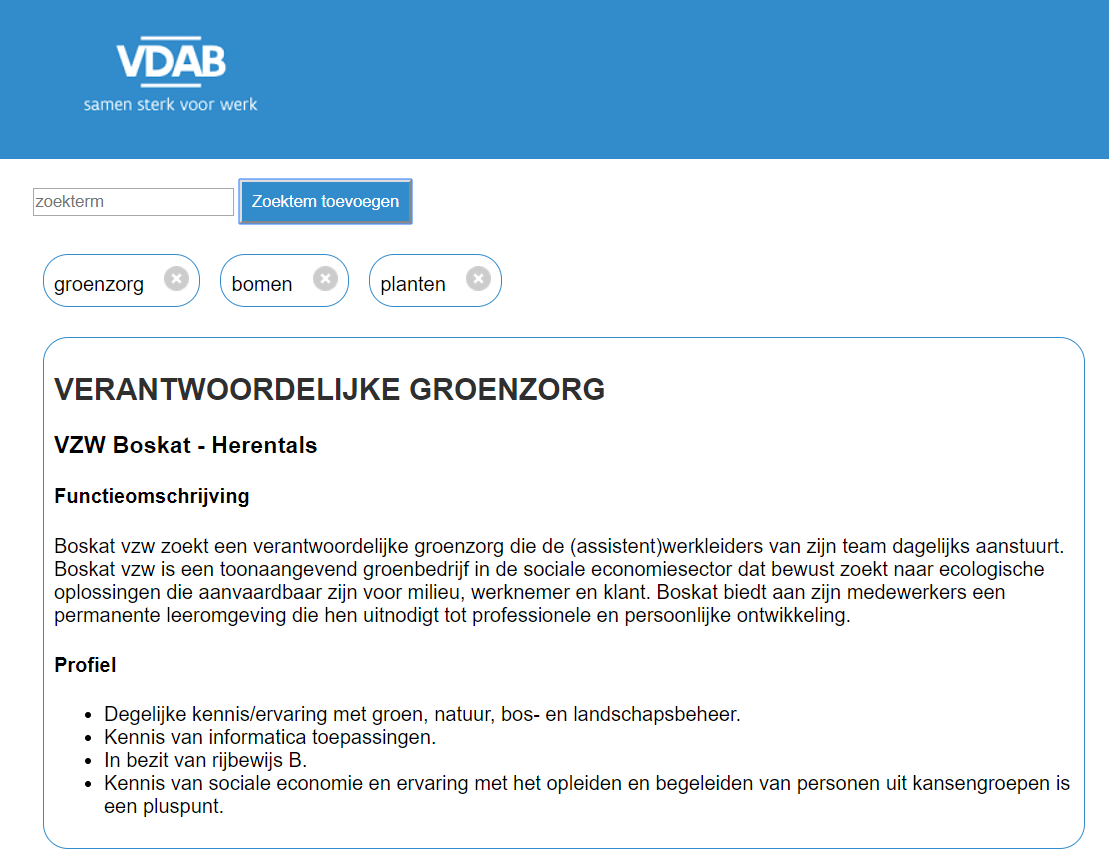
# Oefeningen DOM

## Oefening 1 *VDAB* [Examen Voorjaar 2017]

De werking van de webpagina is als volgt

* Je geeft een zoekterm in in het tekstvak met id zoekterm. Je klikt op de knop Zoekterm toevoegen =>
  + De zoekterm wordt toegevoegd aan het lijstje bovenaan indien deze nog niet bestaat.
  + Alle vacatures waarvan de **titel** één of meerdere van de zoektermen bevat, worden getoond. Het zoeken is niet hoofdletter gevoelig
* Als bij een zoekterm op  geklikt wordt, verdwijnt de zoekterm uit het lijstje bovenaan en wordt de lijst van vacatures bijgewerkt.
* Als de pagina geladen wordt, moeten de laatst gekozen zoektermen onmiddellijk verschijnen en de overeenkomstige vacatures worden onmiddellijk getoond.



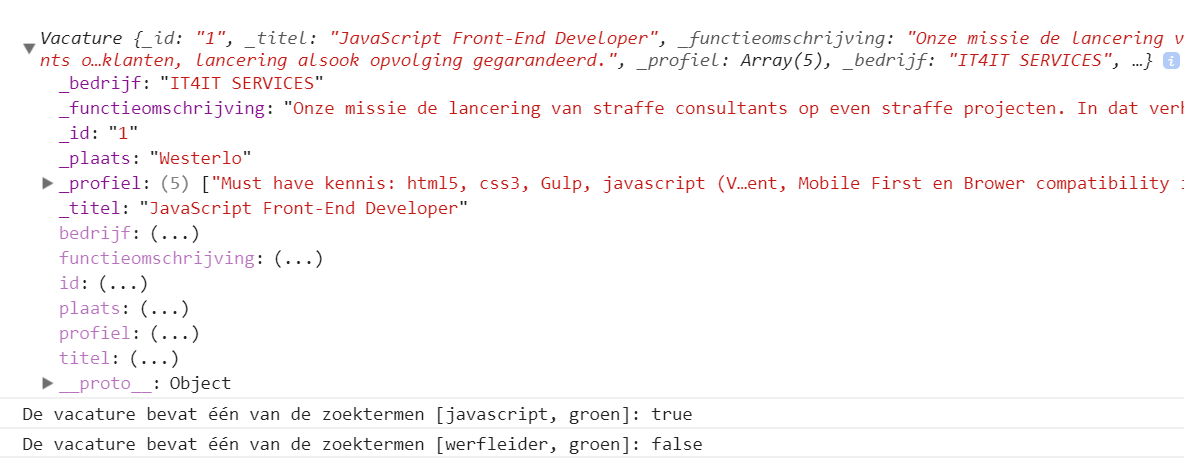
**Deel 1**

Voorzie een klasse Vacature. Elke vacature heeft id, titel, functieomschrijving, profiel, bedrijf en plaats.

Voorzie getters en setters

Voorzie een functie bevatZoekterm(zoektermen). Deze functie heeft retourneert true als de titel van de vacature één of meerdere van de zoektermen (dit is een array) bevat. Dit mag niet hoofdlettergevoelig zijn.

Test de code door in de init – functie de testcode voor Deel 1 uit commentaar te zetten. Zet de testcode vervolgens weer in commentaar.



**Deel 2**

Zet de bestaande code voor VacatureRepository uit commentaar.

Implementeer de methode voegVacatureToe om één vacature toe te voegen aan de array van vacatures. Maak gebruik van de meegegeven parameters.

Test de code door in de init – functie de testcode voor Deel 2 uit commentaar te zetten. De 4 vacatures moeten uitgeschreven worden. De undefined is afkomstig van de filter – functie die momenteel nog niets retourneert.



Implementeer de methode filter die de vacatures retourneert die voldoen aan één of meerdere van de meegegeven zoektermen.

Controleer of de testcode de gewenste resultaten geeft. Zet de testcode voor Deel 2 vervolgens weer in commentaar.



**Deel 3**

Zet de bestaande code voor VdabComponent uit commentaar.

Zet de bestaande testcode voor Deel 3 uit commentaar.

Voeg code toe aan de init – functie zodat het volgende gebeurt als op de knop Zoekterm toevoegen geklikt wordt

* + Er wordt gecontroleerd of deze zoekterm nog niet voorkomt. Er wordt daarbij geen onderscheid gemaakt tussen hoofdletters en kleine letters.
  + Als de zoekterm al voorkomt => Er verschijnt een alert met “Deze zoekterm bestaat al”
  + Anders wordt de zoekterm in kleine letters toegevoegd aan de array zoektermen
  + Het tekstvak wordt leeg gemaakt
  + Alle zoektermen worden getoond in de console

Test deze code!



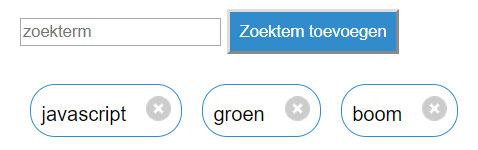
Implementeer de methode zoektermenToHtml om de zoektermen bovenaan te laten zien.

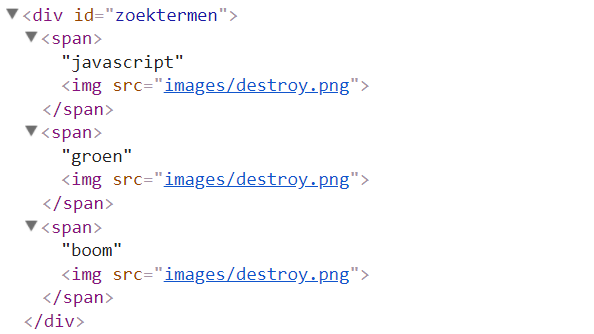
* + De onderstaande html code wordt dynamisch gegenereerd. Wat in een kadertje staat, verschilt per zoekterm



* + Als op  wordt geklikt, wordt de zoekterm verwijderd en wordt het resultaat aangepast
  + Op het einde van de methode wordt de functie vacaturesToHtml opgeroepen

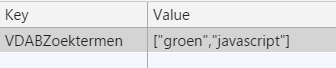
Test deze code door in de init – functie deze functie aan te roepen als op de knop Zoekterm toevoegen geklikt wordt





Implementeer de methode getZoektermenFromStorage om de zoektermen op te vragen uit de storage.

Als de storage een sleutel VDABZoektermen bevat, moet de array zoektermen opgevuld worden met de overeenkomstige waarden uit de storage



Implementeer de methode setZoektermenInStorage om de zoektermen op te vragen uit de storage.

Deze functie wordt gebruikt om de array zoektermen weg te schrijven naar de storage. De gebruikte sleutel is VDABZoektermen.

Denk goed na waar je de methode setZoektermenInStorage moet oproepen (2 plaatsen). Je moet er altijd voor zorgen dat telkens wanneer de array zoektermen gewijzigd wordt, de storage gesynchroniseerd wordt: Dus zowel wanneer er een zoekterm toegevoegd wordt, als wanneer er één verwijderd wordt

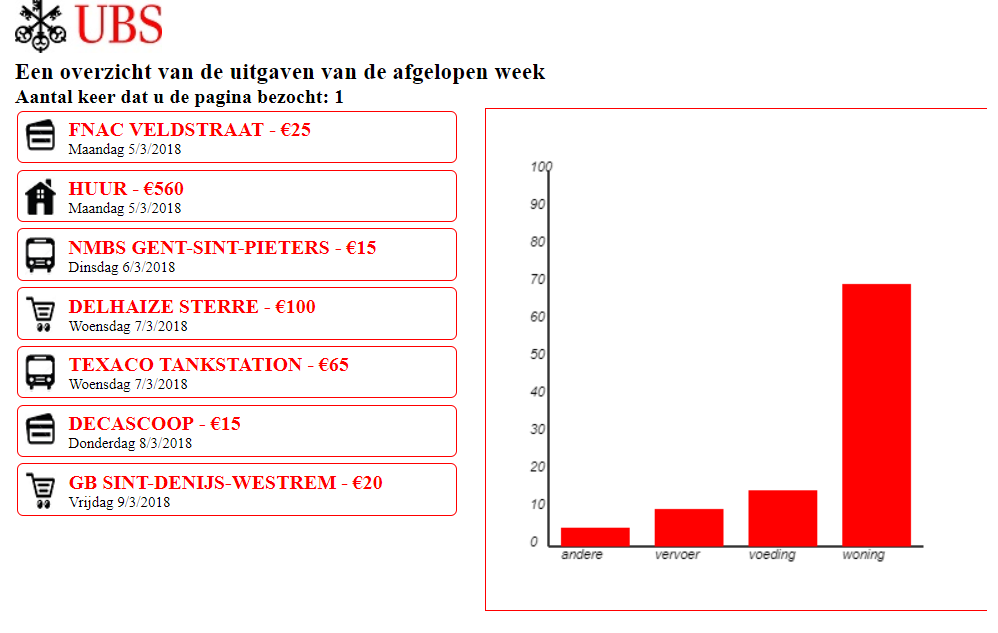
Roep de methode getZoektermenFromStorage op in de init – functie, gevolgd door een oproep van de functie zoektermenToHtml zodat wanneer de pagina geladen wordt, de zoektermen geladen en weergegeven worden.

## Oefening 2 Bank [Examen Voorjaar 2016]

Op de officiële KBC website kan je een overzicht krijgen van je uitgaven verdeeld over verschillende categorieën. De bedoeling van deze oefening is dat we een sterk vereenvoudigde versie hiervan maken.

Aan de linkerkant verschijnt een overzicht van alle uitgaven. Aan de rechterkant maken we gebruik van een canvas om een kolomgrafiek te laten zien. Hierop wordt de **procentuele** verdeling van de uitgaven uitgezet. In dit voorbeeld zijn er 4 categorieën: andere, vervoer, voeding en woning

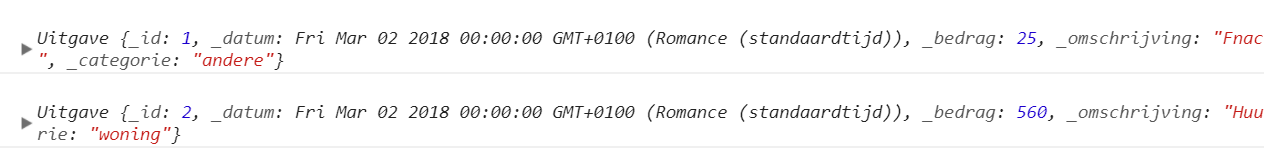
* totaal bedrag van de uitgaven = 800 EUR
* categorie: andere = 40 EUR => 5% van het totaal bedrag
* categorie: vervoer = 80 EUR => 10% van het totaal bedrag
* categorie: voeding = 120 EUR => 15% van het totaal bedrag
* categorie: woning = 560 EUR => 70% van het totaal bedrag



**Deel 1**

Voorzie een klasse Uitgave. Elke uitgave bestaat uit een id, datum, bedrag, omschrijving en een categorie. Voorzie getters en setters

Test de code door in de init – functie de testcode voor Deel 1 uit commentaar te zetten. Zet de testcode vervolgens weer in commentaar.

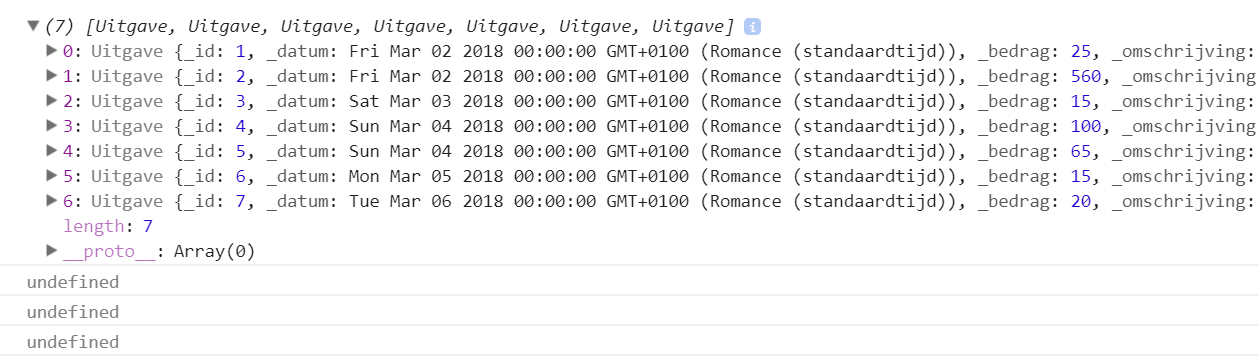


**Deel 2**

Zet de bestaande code voor UitgavenRepository uit commentaar.

Implementeer de methode voegUitgaveToe om één uitgave toe te voegen aan de array van uitgaven. Maak gebruik van de meegegeven parameters.

Test de code door in de init – functie de testcode voor Deel 2 uit commentaar te zetten. De 7 uitgaven moeten uitgeschreven worden. De undefined zijn afkomstig van de functies die momenteel nog niets retourneren.



Implementeer de methode geefCategorieen die een alfabetisch gesorteerde array van de unieke categorieën retourneert.

Controleer of de testcode het gewenste resultaat geeft.



Implementeer de methode totaalBedragUitgaven die het totale bedrag van de uitgaven retourneert.

Controleer of de testcode het gewenste resultaat geeft.



Implementeer de methode uitgavenPerCategorie die het totale bedrag van de uitgaven voor de opgegeven categorie retourneert.

Controleer of de testcode de gewenste resultaten geeft. Zet de testcode voor Deel 2 vervolgens weer in commentaar.



**Deel 3**

Zet de bestaande code voor BankComponent uit commentaar.

Zet de bestaande testcode voor Deel 3 uit commentaar.

Bekijk de implementatie van de functie datumNotatie. Deze functie zorgt ervoor dat een datum wordt uitgeschreven zoals in het bovenstaande voorbeeld, bijvoorbeeld: Maandag 8/3/2018

Implementer de functie getAantalBezoekenFromStorage die wordt gebruikt om het aantal keer dat de website werd bezocht op te halen uit te storage. Maak gebruik van de property aantalBezoeken . Roep deze functie op in de init – functie op de juiste plaats.

Implementer de functie setAantalBezoekenInStorage die wordt gebruikt om het aantal keer dat de website werd bezocht weg te schrijven naar de storage. Maak gebruik van de property aantalBezoeken Roep deze functie op in de init – functie op de juiste plaats.

Implementeer de functie tekst waarin de onderstaande html code dynamisch wordt gegenereerd. Per categorie komt er een andere afbeelding: andere.png / vervoer.png / voeding.png / woning.png. De omschrijving staat altijd in hoofdletters.

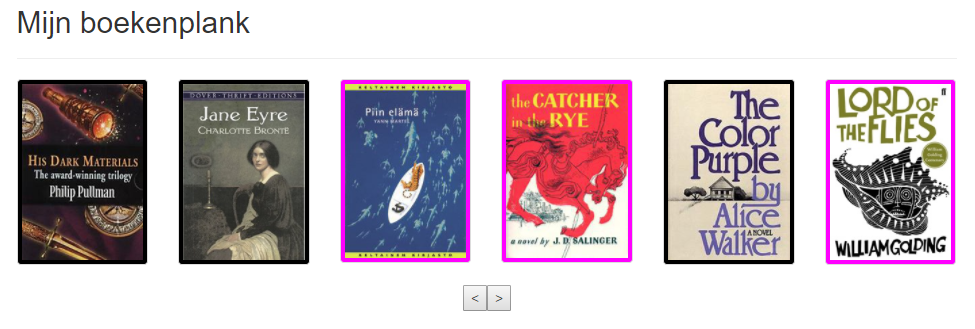


Voor elk gegeven uit de repository

## Oefening 3 Boeken [Examen 2de zit 2017]

De webpagina bevat een overzicht van je boeken. Sommige boeken heb je reeds gelezen en worden getoond in een roze kader. De boeken die je nog niet las staan in een zwart kader. Wanneer je op een boek klikt dat je nog niet las verandert de kleur. Wanneer je klikt op een reeds gelezen boek verandert er niets. De boeken die je reeds las, worden bijgehouden in de storage. Zo krijg je telkens het correcte overzicht wanneer je de webpagina opent.

Op je boekenplank zie je telkens maar 6 van je boeken. Onder de boeken bevindt zich een balk voor de navigatie. Door te klikken op een pagina krijg je de gepaste boeken te zien.



*De derde pagina van de boekenplank, bevat boeken 13 tem 18 van alle boeken.*

*De gelezen boeken staan in een roze kader.*

**Deel 1 – De class Boek**

Een boek heeft een *id*, een *titel* een *afbeelding*. De klasse is reeds geïmplementeerd.

class Boek {

constructor(id, titel, afbeelding) {

this.id = id;

this.titel = titel;

this.afbeelding = afbeelding;

}

get id() {return this.\_id;}

get titel() {return this.\_titel;}

get afbeelding() {return this.\_afbeelding;}

set id(value) {this.\_id = value;}

set titel(value) {this.\_titel = value;}

set afbeelding(value) {this.\_afbeelding = value;}

}

**Deel 2 – De boekenrepository**

Een boekenrepository bevat een lijst boeken. De constructor maakt een nieuwe lijst aan en vult deze meteen op met boeken. Dit is reeds geïmplementeerd. Merk op dat de boeken **niet** op alfabetische volgorde van titel in de lijst zitten.

class BoekenRepository {

constructor() {

this.boeken = [];

this.boekenVullen();

}

get boeken() {return this.\_boeken;}

set boeken(value) {this.\_boeken = value;}

voegBoekToe(id, titel, afbeelding) {

this.\_boeken.push(new Boek(id, titel, afbeelding));

}

boekenVullen() {

this.voegBoekToe(1, "Wuthering Heights", "WutheringHeights.jpg");

this.voegBoekToe(2, "Gullivers Travels", "GulliversTravels.jpg");

…

}

}

Aan een boekenrepository kan je een aantal boeken vragen. Deze functie moet je zelf implementeren. Je geeft aan vanaf het hoeveelste boek en hoeveel boeken je wenst. In de afbeelding hierboven werden voor de derde pagina 6 boeken gevraagd vanaf boek 13.

geefBoeken(vanafBoek, aantalBoeken) {

…

}

**Deel 3 – BoekenComponent - storage**

In de component wordt een lijst met de id’s van de gelezen boeken bijgehouden in de variabele gelezenBoeken. De functie voegGelezenBoekToe is reeds gegeven.

class BoekenComponent {

constructor(window) {

this.boekenRepository = new BoekenRepository();

this.gelezenBoeken = []; // bevat de id's van gelezen boeken

// bevat het nummer van de pagina die momenteel getoond wordt

this.actievePagina = 1;

this.storage = window.localStorage;

this.aantalBoekenPerPagina = 6;

}

// getters

// setters

}

Implementeer de 2 functies om de gelezen boeken bij te houden in en op te halen uit de storage. Zorg dat de property gelezenBoeken correct wordt weggeschreven/opgehaald.

// getGelezenBoekenFromStorage haaltde lijst met id's van gelezen boeken

// op uit de storage

getGelezenBoekenFromStorage() {…}

// setGelezenBoekenInStorage plaatst de lijst van id's van gelezen boeken

// in de storage

setGelezenBoekenInStorage() {…}

**Deel 4 – toHTML**

Implementeer nu de functie **boekenToHtml** verder. Deze functie genereert de boekenplank. Deze bevat 6 boeken voor de actievePagina (enkel op de laatste pagina kunnen er eventueel minder dan 6 boeken staan). Zorg dat wanneer er **geklikt** wordt op een nog niet gelezen boek de functie voegGelezenBoekToe wordt aangeroepen met de juiste parameter. Gelezen boeken **reageren niet** op een klik.

Hieronder zie je de HTML die je moet genereren binnen de div met id “boeken”



id van boek

titel van boek

afbeelding van boek