W1D4

**Leerdoelen:**

* **Ik kan uitleggen wat de Document Object Model is**

Je kunt de DOM zien als de stappen die de browser neemt nadat hij het HTML-document heeft ontvangen om de pagina te renderen.De eerste informatie die de browser ontvangt, is het HTML-document. Hierna begint het de inhoud van dit document te ontleden en extra bronnen op te halen, zoals afbeeldingen, CSS- en JavaScript-bestanden.

De DOM is wat wordt gevormd door deze ontleding van informatie en bronnen. Men kan het beschouwen als een gestructureerde, georganiseerde versie van de code van de webpagina.

***De DOM is wat u ziet wanneer je “Inspecteren” of “Element Inspecteren” in je browser aanklikt***



De browser zelf gebruikt het DOM om allerlei regels rond vormgeving en andere informatie op de HTML toe te passen terwijl het de pagina weergeeft.   
Ontwikkelaars zoals jij kunnen Javascriptcode schrijven die het DOM manipuleert nadat de pagina al is opgeladen in de browser.

We hebben een eenvoudig voorbeeld gemaakt op [dom-example.html](https://github.com/mdn/learning-area/blob/master/javascript/apis/document-manipulation/dom-example.html)   
Het is een zeer simpele pagina die een[<section>](https://developer.mozilla.org/nl/docs/Web/HTML/Element/section) element bevat waarin je een afbeelding en paragraaf kan vinden. In de paragraaf zit ook nog een link. De html-broncode ziet er zo uit :

<!DOCTYPE html>

<html>

<head>

<meta charset="utf-8">

<title>Simple DOM example</title>

</head>

<body>

<section>

<img src="dinosaur.png" alt="A red Tyrannosaurus Rex: A two legged dinosaur standing upright like a human, with small arms, and a large head with lots of sharp teeth.">

<p>Here we will add a link to the <a href="https://www.mozilla.org/">Mozilla homepage</a></p>

</section>

</body>

</html>

Het DOM daarentegen ziet er zo uit:



* **Ik kan uitleggen hoe ik de DOM kan gebruiken om iets op een website te veranderen**

<https://developer.mozilla.org/nl/docs/Learn/JavaScript/Client-side_web_APIs/Manipuleren_documenten>

//Om een element in het DOM te manipuleren, moet je het eerst selecteren

//en een verwijzing ernaar opslaan in een variabele. Om dat te doen,

//voeg je de volgende regel aan je script toe:

var link = document.querySelector("a");

//Nu we een verwijzing naar het element "a" hebben opgeslagen in een variabele,

//

//kunnen we beginnen met een aantal eigenschappen en methodes die beschikbaar

//zijn gemaakt voor de manipulatie van het DOM. (Deze worden gedefiniëerd in

//interfaces zoals het HTMLAnchorElement voor een <a> element, zijn

//algemenere ouder-interface, het HTMLElement, en Node die alle knooppunten

//in het DOM vertegenwoordigt.

//Laat ons eerst en vooral de tekst in de hyperlink

//veranderen door de waarde van de Node.textContentaan te passen. Voeg de volgende

//regel toe aan de vorige:

link.textContent = "Mozilla Developer Network";

//We moeten ook de URL veranderen waar de hyperlink naar verwijst zodat die ons niet

//naar de verkeerde pagina brengt als we erop klikken. Voeg daarvoor de volgende regel toe: :

link.href = "https://developer.mozilla.org";

//Opmerking : Zoals met zoveel in Javascript, zijn er vele manieren om een element te selecteren

//en om een verwijzing ernaar op te slaan in een variable.

//Document.querySelector() is de aanbevolen moderne manier.

//Dat komt goed uit want deze methode laat je elementen selecteren met CSS-selectoren.

//De regel hierboven komt overeen met het eerste <a> element dat in het document opduikt.

//Als je meer dan één element wil manipuleren, kan je Document.querySelectorAll() gebruiken.

//Deze methode selecteert elk element in het document dat overeenkomt met de selector en slaat

//die op in een array.

* **Ik kan uitleggen hoe ik de DOM kan gebruiken om iets op een website te veranderen**

**–** Change / remove HTML elements in de DOM / on the page

– Change and add CSS styles to elements

– Read and change element attributes (href, src, alt, custom)

– Create new HTML elements and insert them into the DOM / the page

– Attach event listeners to elements (click, keypress, submit)

**Leerdoelen:**

* **Ik kan met de console in de Chrome Developer tools:** 
  + **kleine stukjes JavaScript uitvoeren done**
  + **met behulp van JavaScript interacteren met de DOM done**

**Leerdoelen:**

* **Ik kan een element selecteren by ID**

1 element in pagina nodig om te updaten

2 getElementByID = element heeft een id met naam

bv <div id = “page-banner”></div>

elke id is uniek dus pak je id page-banner, dan is er daar maar 1 van

hoe:

a- document  
 b-use method on this document dus

c- document.getElementByID

dan parapeter('')

d- document.getElementByID('page-banner')

e-nu onderstaand elemant terug gekregen

****

f -nu hebben we dat element te pakken

**We kunnen het ook in een variabele stoppen, zodat we het later kunne gebruiken**

hoe:

a- var banner = document.getElementByID('page-banner')

b- roep banner op en druk je nu op enter dan kan je zien dat het element   
 nu in de variabele zit (zie onder)



dus: roepen we nu banner op dan zit daar de div-element in welke we kunnen gaan aanpassen/stylen

* **Ik kan een element selecteren by Class**

****

We kunnen nu dus alle class pakken (meerdere elementen) en hier weer iets mee doen.

Hoe:

a- document.getElementByClassName('title')

b- in een variabele zetten zodat we er later iets mee kunnen doen

var titles = document.getElementByClassName('title')

zie vb lijkt op een array, maar dit is het nog niet



we kunnen er wel elementen uit halen, zie vb  
type in titles[0]

dan wordt het eerste element terug gegeven nl

<h1 class='title'>Bookorama</h1>

* **Ik kan een element selecteren by Tag**

dus bij een collectie van elemanten met allemaal dezelfde tag in onderstaand voorbeeld dus

<li></li> tags



we stoppen het in een variabele;

var lis = document.getElementByTagName('li')

zie vb

****

**We kunnen nog wel bij de individuele elementen door het volgende in te vullen**lis[0] of lis[2]

****

**Maar wanneer we in de DOM zitten willen we iets doen met de elementen die we terug krijgen**

**vb  
query de DOM voor alle title class elements en ze vervolgens stylen, dan moeten we een manier vinden om door al die elementen te loopen, het kunnen er nl heel veel zijn.**

**Dit kan met een for loop en we kunnen daarna met elk title class iets doen**

**zie vb en inplaats van loggen kunnen we dus stylen of attribuut toevoegen etc  
**

**maar er is nog een manieren**

**eerst een variabele aanmaken**

var titles = document.getElementByClassName('title')

dit is eigenlijk een array, en in een array kunnen we de methode forEach gebruiken

bv

–-hier is het item, het item dat steeds terug komt als we door de array loopen

titles.forEach(function(item)) {

console.log(item);

}

zie vb



dan error zie vb



error, omdat titles geen array is , maar wat we kunnen doen is er een array van maken

eerst: console.log(array.isArray(titles));

als we dit loggen op de console krijgen we false terug als het geen array is en true als het wel een array is



false dus geen array  
  
maar we kunnen er wel een array van maken

console.log(array.isArray(titles));

console.log(array.isArray(array.from(titles)));

wat we hier zeggen is dat we een array willen maken van titles



wanneer we dit loggen op de console zou er true uit moeten komen



nu kunnen we forEach wel gebruiken

titles.forEach(function(item)) {

console.log(item);

}



dus

Array.from(titles).forEach(function(item) {

console.log(item);

})



Array.from(titles)geeft een array terug

* **Ik kan met document.queryselector werken.**

Met een query wordt een opdracht bedoeld die aan een database wordt   
gegeven om een bepaalde actie uit te voeren, die ook potentieel gegevens teruggeeft.   
– [querySelector()](https://developer.mozilla.org/nl/docs/Web/API/Document/querySelector)heet, Met deze functie kan je de verwijzing opslaan in een variabele . Als u iets met een element op uw pagina wilt doen, zult u het eerst moeten selecteren en opslaan in een variabele.

is de makkelijkste manier om een element uit de DOM te halen

voorbeeld

We willen dus wrapper uit de DOM halen, dit doen we door  
document.querySelector('#wrapper); dit stoppen we in een variabele dus,

const wrap = document.querySelector('#wrapper');

console.log(wrap);

zie vb, nu is het gelogt naar de console, dit betekend dat we het element wrapper hebben gepakt

volgende voorbeeld: we willen **the wise man's fear** pakker, zie vb

wanneer we dit inspecteren op de console zien we dat het een:

-span is -met class of name -li tag – second child in ul -in book-list

volgorde

1 book-list id

2 li, maar wel de second child li dus  
 li:nth-child(2)

3 class of name = .name (is een span-tag)

dus

('#book-list li:nth-child(2).name');

document.querySelector

const wmf (wise men's fear) =

const wmf = document.querySelector('#book-list li:nth-child(2).name');



nu loggen we dit naar console

nu loggen we dit naar de console:

const wmf = document.querySelector('#book-list li:nth-child(2).name');

console.log(wmf);

zie vb



nu zien we het element in de console.

Nu hebben we 1 element gepakt, maar we willen alle elementen (span-tags pakken, dat doen we zo:



var books = document.querySelector('#book-list li .name');  
Hier staat dus dat we met de querySelector uit de;  
#book-list  
alle li tags, met class of name  
dus  
alles met class of name – in een li – in de book-list  
dit loggen we naar de console

var books = document.querySelector('#book-list li .name');  
console.log(books);



Wat is er nu gebeurd, alleen **Name of the Wind naar console gelogt**, waarom??  
Nou, wanneer we de **querySelector** gebeuren geeft deze alleen maar **1 element** terug en dus niet meerdere elementen,  
hoe dan wel:

var books = document.querySelector**All**('#book-list li .name');  
console.log(books);

dus document.querySelector – geeft 1 element terug en  
 document.querySelectorALL – geeft eencollectie van elementen teruggeeft  
vb  
var books = document.querySelectorAll('#book-list li .name');  
console.log(books);



nu zien we de collectie van elementen in de console.log

Om gebruik te maken van de collectie var = books, kunnen we er een array van maken zodat we bv de methode forEach kunnen gebruiken (for loop kannatuurlijk ook)

eerst een array maken van books

|  |  |
| --- | --- |
| Array.from(books) | hier staat, we willen van de variable books een array maken |
| Array.from(books)**.forEach** | dan wanneer het een array is kunnen we bv forEach gebruiken |
| Array.from(books).forEach**(function)** | We willen er iets mee doen, en dit doen we met een function |
| Array.from(books).forEach(function**(book)**) | In de functie willen we door de book parameter gaan (ittereren) |
| Array.from(books).forEach(function(book)) **{**  **console.log(book);**  **})** | Console.log  zie vb |





Nu zien we ze gelogt naar de console!!!!!

**Leerdoelen:**

* **Ik kan een element selecteren by ID YES**
* **Ik kan een element selecteren by Class of Tag YES**
* **Ik kan met document.queryselector werken. YES**

[**JS: Browser Events**](https://www.notion.so/1060d28629a541f6a39ad2e385b1edbf)

**info**

[**https://developer.mozilla.org/nl/docs/Web/Events**](https://developer.mozilla.org/nl/docs/Web/Events)

**How to remove content from the DOM and eddit content on the DOM**



|  |  |
| --- | --- |
| Zie vb geselecteerd <div id “book-list”></div>  wanneer ik een click event wil geven aan de titel Books to Read dan moet ik daar een eventlistener aan hangen  ook maken we een call back function, die zodra het event gebeurt  dus nu maken we een eventlistener aan Books to Read h2 | |
| 1  var h2=document.querySelector('#book-list h2') | Query the DOM in de book-list id -nu hebben we de h2 te pakken |
| 2  h2.addEventListener('click') | Nu willen we er een eventlistener aan toe voegen aan het element h2, in dit geval een  click event  -de methode is addEventListener -de parameter is 'click' , dit kan dus vanalles zijn zie lijst |
| h2.addEventListener('click', **function() {**  **})** | Volgende parameter is een callback function  deze callbackfunction wordt actief zodra dit click event gebeurd op de pagina  dus  wanneer iemand op het h2 element clickt (dit is dus Books to Read)  ontvangt de Eventlistener dit en voert de callback function uit als reactie |
| h2.addEventListener('click', function(**e**) {  }) | Deze callback funtion gaat door een parameter  **e** en die kan je elke naam geven (hier dus e) |
| h2.addEventListener('click', function(e) {  console.log(e**.target**);  }) | Nu console.log(e.target)  target is een propperty on the event opbject  en vertelt ons op welk element wordt er geklikt, dus wat is de target van de event |
| h2.addEventListener('click', function(e) {  console.log(e.target);  **console.log(e);**  }) | Console.log(e); zodat we dit kunnen zien in onderstaand voorbeeld |
|  | **Nu hebben we een eventlistener aan de h2 toegevoegd** |



Dus als we er nu op klikken krijgen we  
1 h2 en dan,  
2 de event ,zie vb



We kunnen bijvoorbeeld zien dat de shiftkey niet is ingedrukt.

**Nu gaan we eventlisteners toevoegen aan de 4 delete knoppen**

****

|  |  |
| --- | --- |
| 1 query de DOM om alle 4 de delete knoppen te pakken  2 ze zitten allemaal in de 4 li tags  3 en ze hebben allemaal een class of delete | |
| Var btns = document.querySelectorAll('#book-list.delete') | |
| Variabele btns  documentquerySelectorAll  #book-list  .delete | Variabele buttens  we willen ze allemaal pakken  ze zitten in #book-list  ze hebben allemaal class delete |
| Nu hebben we een referentie naar de collectie en die zitten allemaal in de variabele |  |
| Nu willen we door die collectie lopen en aan elk delete element een eventlistener hangen |  |
| Het is nu nog een collectie van elementen dus nog niet 1 element | Btns.addEventListener kan dus niet |
| We moeten door de collectie heen loopen en aan elk event een eventlistener hangen | Dus array maken **Array.from( )** en forEachmethode aanhangen |
| Array.from(btns).forEach(function(btn)) {  btn.addEventListener('click', function(e)) {    const li = e.target.parentElement; | Hier pakken we elke keer de button als we ittaraten  Hier plakken we een eventListener aan elke button (een click event) + een callback function wanneer er een event is op deze eventobject  we willen de li tag deleten want daar zit ook de naam is  Hier hebben we de li te pakken en deze is gekoppeld aan elke button waarop we klikken  Nu willen we de li deleten |
| parent.removeChild(li) | Dit is de manier waarop we in javascript iets deleten |
| const **li** = e.target.parentElement; | Deze li willen we deleten,maar we moeten eerst de parent hebben en dit doen we door |
| **li.parentNode**.removeChild(li) | Eerst pakken we de parentNode van de li  dat is de ul |

****

|  |  |
| --- | --- |
| li.parentNode.**removeChild**(li)  li.parentNode.removeChild(**li**) | Dan zeggen we  van deze ul willen we een child removen  en de child we want to remove is  li |
| dus | |
| Array.from(btns).forEach(function(btn)) {  btn.addEventListener('click', function(e)) {  const li = e.target.parentElement;  li.parentNode.removeChild(li)  });  ]); |  |

****

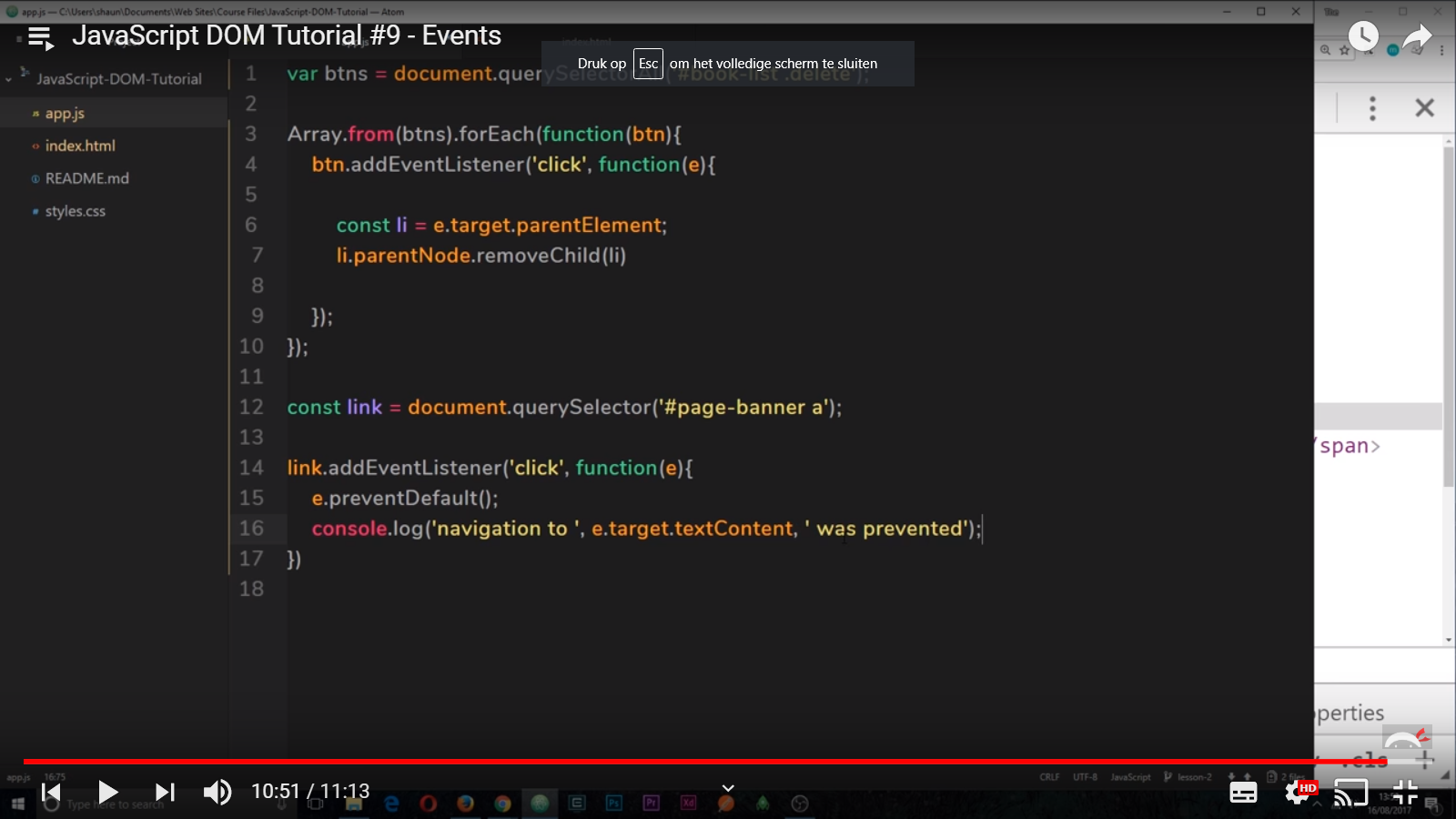
**Dus wanneer we nu iets deleten zien we dat het wordt verwijderd van de pagina  
Now we have succesfully reacted to that click event, navigating on the DOM to the li  
and deleted it from the ul**

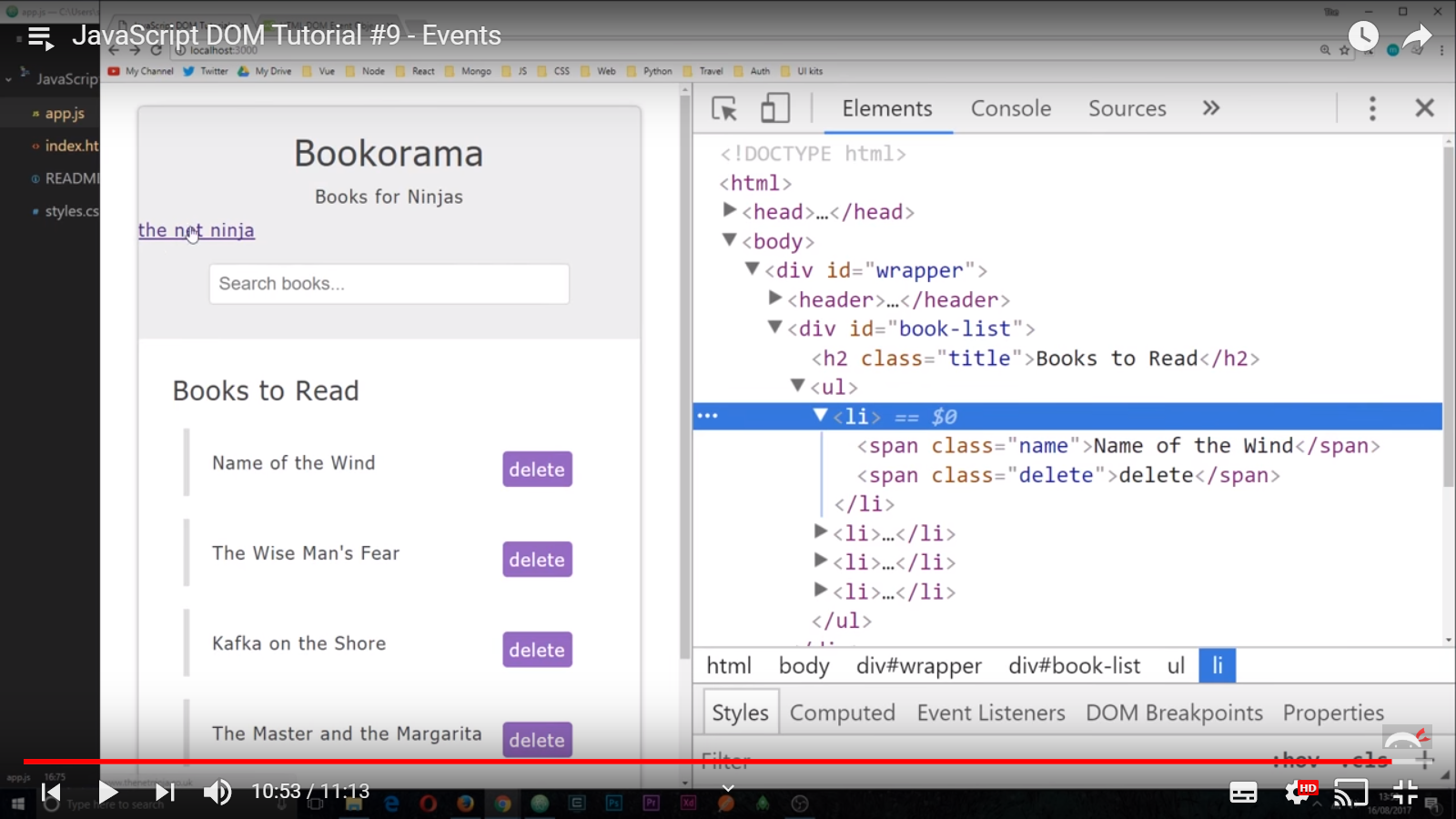
**Prevent default behaviour from an a tag  
default behaviour from an ankor tag is to go to the webpage on which it is linked to**

**how to prevent this vb (click here)**

****

|  |  |
| --- | --- |
|  | 1 query the DOM |
| Const link = document.querySelector('#page-banner a') | |
| Link.addEventListener('click', function(e){  e.preventDefault();  console.log('navigation to ', e.target.textContent, 'was prevented');  }) | 2 add eventListener to this link met een click event en een callback function  deze methode op de event object is gonna prevent de default action ,  which is to navigate to the other website  merk op; de text content is dat wat er in de Htm staat, en daar staat nnu nog click here, dus dat kan je aanpassen |

****

Click on the net ninja, dan

****

en worden dan dus niet doorgetuurd