Forms

Forms herhaling

Forms

- Formulieren worden gebruikt om **input** te **verzamelen**
- De input moet behandeld worden door een front-end of back-end script
  - Anders gebeurt er niets met de ingevulde data
- In html definiëren we een formulier aan de hand van een form element
  - Een webpagina kan meerdere form elementen bevatten.

Form element

Een form element kan 4 attributen bevatten:

- Action => naar waar de formulier data gestuurd moet worden
- Method => hoe de data verstuurd moet worden
- Autocomplete => het toelaten van autocomplete door de browser
- Novalidate => aangeven of de data gevalideerd moet worden

[codepen]

Method

GET

- Wordt als naam/value paar toegevoegd aan de URL
- Max url lengte is 2048 karakters
- Kan gebookmarked worden
- Kan gebruikt worden voor query strings, niet gevoelige data

POST

- Plaatst de formulier data in de body van de HTTP request
- Geen grootte beperking
- Post submission kan niet gebookmarked worden
- Veiliger

[codepen]

Form elementen

Er zijn heel wat form elementen. De belangrijkste zijn:

- input - w3schools - MDN
- label - w3schools - MDN
- select - w3schools - MDN
- textarea - w3schools - MDN
- button - w3schools - MDN

Meer elementen kan je vinden op w3schools of MDN

[codepen]

Speciale form attributen

- For
  - Dit attribuut zorgt ervoor dat wanneer je op een label klikt, het juiste input element gefocust is. Dit moet steeds verwijzen naar een id van het element dat je wil focussen.
- Name
  - De name van een input element wordt gebruikt om de data te labelen wanneer deze wordt doorgestuurd via een form
- Value
  - De waarde van een element

Formulieren uitlezen

Formulieren uitlezen

We kunnen adhv javascript op 2 manieren data uit een formulier ophalen:

1. Aan de hand van DOM selectie
2. Aan de hand van het forms object

Forms object

In javascript kunnen we via het document object alle formulieren van een webpagina ophalen. => document.forms

Het formsobject bevat een dictionary met alle formulieren op deze pagina.
1 formulier object kan je aanspreken aan de hand van het id van dit form of met behulp van de arraynotatie.

[codepen]

Forms data uitlezen

Met behulp van de value eigenschap kan je de data uit een formulier element halen.

[codepen]

Preventdefault

preventDefault is een methode die er voor zorgt dat de **default actie van een form niet uitgevoerd wordt**.

Met andere woorden de form wordt niet verzonden naar de action url.

PreventDefault is een methode van het eventobject

[codepen]

StopPropagation

Propagation is het 'doorgeven' van een event. Wanneer er zowel op een element als op zijn parent een eventlistener staat, wordt dit event doorgegeven.

Met stopPropagation kunnen we dit gedrag tegen gaan.

Let op in combinatie met useCapture, moeten we goed in het oog houden wat we aan het doen zijn.

[codepen]

Form validatie

Form validatie

Er zijn verschillende manieren om aan form validatie te doen.

- HTML form validatie
- JS form validatie

Html form validatie

Kunnen we doen door:

- Input type te specifiëren (date, email, number)
- Gebruik te maken van html attributen

Html form validatie attributen

- max ⇒ maximum waarde van een element
- min ⇒ minimum waarde van een element
- pattern ⇒ patroon waaraan voldaan moet worden
- required ⇒ specificeert of een element verplicht in te vullen is.

[codepen]

CSS validatie pseudo-elementen

- :disabled ⇒ elementen die disabled zijn
- :invalid ⇒ elementen met invalid gegevens
- :optional ⇒ optionele elementen (niet required)
- :required ⇒ verplichten velden
- :valid ⇒ valid elementen

[codepen]

JS validatie

Wanneer we met html validatie niet het gewenste effect krijgen kunnen we ook met behulp van js valideren.

Met js kunnen we:

- Foutboodschappen anders tonen (met behulp van dom selectie en manipulatie)
- Valideren op andere criteria

Validity element

Elk forms element heeft een validity element. Hiermee kunnen we controleren of een html element voldoet aan de html validatie.

[codepen]

Custom validatie messages

Het is ook mogelijk om gebruik te maken van de built-in validatie met custom messages

Dit kunnen we doen met behulp van de methode: **setCustumValidity**

Met de methode **reportValidity** kunnen we deze tonen

[codepen]

Events

Feedback op foutieve data wordt best asap gegegeven

Enkele interessante events in combinatie met validatie zijn:

- Input => wordt aangeroepen wanneer de input veranderd
- Focus => wordt aangeroepen wanneer een element de focus krijgt
- Blur => wordt aangeroepen wanneer een element de focus verliest.

[codepen]

RegEx

RegEx is een afkorting voor reguliere expressie

Met reguliere expressies kunnen we patronen valideren zoals met het html attribuut pattern

Zelf een regex schrijven

Een regex wordt steeds tussen 2 forward slashes geschreven //

Aan de hand van symbolen wordt een beschrijving van een mogelijke string gegeven.

- . ⇒ Elke mogelijk karakter behalve een nieuwe lijn
- [a-z] ⇒ Elke karakter tussen a en z
- a* ⇒ 0 of meerdere keren
- a+ ⇒ 1 of meerdere keren
- a? ⇒ 0 of 1 keer
- \ ⇒ escape karakter (bv. \., \+, *)

Meer info op regexr

[codepen]

Data inlezen

Data inlezen

Éen van de manieren om data in te lezen is door middel van **querystrings**.

Dit is het deel dat bij een URL bijkomt wanneer je een formulier via een get request verstuurd.

Met behulp van JavaScript kan je deze uitlezen en gebruiken.

URLSearchParams

Om aan deze querystring te geraken en deze op een eenvoudige manier aan te spreken, kan je gebruik maken van de klasse **URLSearchParams**.

Deze klasse verwacht de querystring uit de url. Deze kan je met behulp van het **window object** verkrijgen.

[codepen]

Extra bron forms

https://web.dev/learn/forms/