LU04b - AJAX Requests

AJAX verwenden

- https://www.w3schools.com/js/js_ajax_intro.asp

Ajax Beispiel „Hallo World“

In einem ersten einfachen Beispiel wollen wir uns die Ajax - Basics einmal anschauen. In diesem Beispiel sende ich einen Request ab, welcher auf eine Textdatei zugreift. Der Inhalt dieser Textdatei wird als String in den `div`-Container „one“ geschrieben.

Du kannst diese kleine Demo unter https://it.bzz.ch/m321/ajaxdemo ausprobieren.

Zunächst einmal der Sourcecode der Dateien, diese gehe ich später Zeile für Zeile durch:

test.txt

```
Hello AJAX-World!
```

index.html

html

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="author" content="Marcel Suter" />
  <meta name="copyright" content="Marcel Suter - BZZ" />
  <meta name="project" content="Simple Ajax" />
  <meta name="description" content="AJAX Text Datei" />
  <title>Textdatei mittels AJAX lesen</title>
  <script src="./ajaxDemo.js"></script>
</head>
<body>
  <div id="one" style="width: 80%; height: 80%; border: dashed
1px;"> </div>
</body>
```

ajaxDemo.js

javascript

```
document.addEventListener("DOMContentLoaded", () => {
  fetch("./test.txt")
    .then(function (response) {
      if (response.ok) {
        return response;
      } else {
        console.log(response);
      }
    })
    .then(response => response.text())
    .then(textData => {
      document.getElementById("one").innerText = textData;
    })
    .catch(function (error) {
      console.log(error);
    });
});
```

Der Aufbau des ersten Ajax Programms

Listener "DomContentLoaded"

```
document.addEventListener("DOMContentLoaded", () => {
  ...
});
```

Dieser Listener wartet darauf, dass der HTML-Code geladen und verarbeitet wurde. Sobald dies der Fall ist, wird der Code in den geschweiften Klammern ausgeführt.

Absenden des Requests

```
  fetch("./test.txt")
```

Mit dem Befehl `fetch` setzen wir einen Request ab. Im einfachsten Fall benötigen Sie nur die URL einer Datei oder eines Services anzugeben.

Auswerten des Resultats

```
  fetch("./test.txt")
   .then(function (response) {
```

```
    if (response.ok) {
      return response;
    } else {
      console.log(response);
    }
  })
  .then(response => response.text())
  .then(textData => {
    document.getElementById("one").innerText = textData;
  })
```

Fetch ist ein asynchroner Befehl, daher definieren wir mit `.then` eine Reihe von Listenern. Sobald der fetch-Befehl fertig ausgeführt wurde, wird der erste `.then`-Listener aktiviert. Im Erfolgsfall werden die weiteren Listener, einer nach dem anderen, durchgeführt.

response.ok

Der erste Listener prüft, ob der Request erfolgreich durchgeführt wurde (`response.ok`). Falls dies zutrifft, wird die Response an den nächsten Listener weitergegeben (`return response`).

response.text()

Da wir „nur“ den Textinhalt der Response benötigen, extrahieren wir diesen Text (`response.text()`).

Anzeige

Im letzten `.then`-Block fügen wir den Text in das div-Element mit der ID one ein.

Fehlerbehandlung

Falls der Request nicht verarbeitet werden kann, wollen wir den Fehler in der Konsole des Browsers ausgeben.

```
fetch("./test.txt")

  ...

  .catch(function (error) {
    console.log(error);
  });
});
```

M321-LU04

Marcel Suter

From:
https://wiki.bzz.ch/ - **BZZ - Modulwiki**

Permanent link:
https://wiki.bzz.ch/modul/m321/learningunits/lu04/request

Last update: **2024/03/28 14:07**