Übung: Erstellung einer Todo-App

Die GUI mit HTML, CSS und JavaScript

In diesem Übungsblatt erstellt ihr eine einfache Webanwendung, mit der ihr tägliche Aufgaben planen und verwalten könnt. Beispiele für diese Art von Anwendungen sind Microsoft Todo oder Zenkit To Do. Die Aufgabe erstreckt sich über mehrere Übungsblätter. Im ersten Teil der Aufgabe erstellen wir eine grafische Benutzeroberfläche mit HTML, CSS und JavaScript.

Vorbereitung

Erstellt für dieses Projekt ein neues Glitch-Projekt. Das Projekt sollte die folgenden drei Dateien beinhalten:

1. `index.html` als Startseite der Anwendung
2. `styles.css` für die Definition des verwendeten Designs
3. `script.js` für die notwendige Programmierlogik der Anwendung.

Die `index.html` befüllt ihr mit dem rudimentären HTML-Template. Die anderen beiden Dateien könnt ihr zunächst leer lassen:

```
<!DOCTYPE html>
<html lang="de">
  <head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width">
    <title>Meine Todo-App</title>
  </head>

  <body>
    <h1>Meine Todo-App</h1>
  </body>
</html>
```

Aufgabe 1: Neue Aufgaben erstellen

Im ersten Schritt sollt ihr die Möglichkeit schaffen, Aufgaben zu erstellen und in einer Liste anzuzeigen. Dazu benötigen wir ein Eingabefeld, in das der Benutzer den Titel der Aufgabe eingeben kann. Gleichzeitig benötigen wir eine Schaltfläche (Button), um die Aufgabe anzulegen.

- Fügt eurer Anwendung ein Texteingabefeld und einen Button hinzu! Gebt dem Button ein sinnvolles Label!
- Stellt sicher, dass beim Klick auf den Button die JavaScript-Funktion `addTask()` aufgerufen wird. Um sicherzustellen, dass der Aufruf funktioniert, soll die Funktion anfänglich ein Lebenszeichen auf der Konsole ausgeben!
- Implementiert nun die Funktion `addTask()`, so dass beim Klick auf den Button das Texteingabefeld ausgelesen wird und eine neue Aufgabe mit dem Inhalt als Titel erstellt wird. Überlegt euch, wie ihr am besten eine Aufgabeliste mit HTML umsetzen könnt und fügt die benötigten Elemente in eure `index.html` ein.
- Leert nach der Erstellung einer Aufgabe das Texteingabefeld, damit der Benutzer direkt die nächste Aufgabe eingeben kann und nicht erst den Titel der vorigen Aufgabe manuell löschen muss!

Aufgabe 2: Aufgaben erledigen

Die Anwendung soll eine Möglichkeit bieten, erledigte Aufgaben als solche zu kennzeichnen und an das Ende der Liste zu sortieren. Darum kümmert ihr euch in diesem Teil der Aufgabe.

- Aufgaben in eurer Liste sollen anklickbar sein. Erweitert eure Anwendung so, dass beim Klick auf eine beliebige Aufgabe die Funktion `taskClick()` aufgerufen wird. Nutzt dazu das Konzept der Event Handler in JavaScript!
- Nachdem eine bestimmte Aufgabe angeklickt wurde, soll diese abgehakt werden. Dazu soll der Titel der Aufgabe durgestrichen werden und der Text leicht ausgegraut werden.
- Wird eine bereits abgehakte Aufgabe erneut angeklickt, so soll sie wieder normal dargestellt werden! Überlegt euch eine Möglichkeit, wie ihr möglichst einfach den Status einer Aufgabe (offen / erledigt) ermitteln könnt.
- Zusätzlich zur visuellen Abhebung sollen erledigte Aufgaben an das Ende der Liste verschoben werden! Erweitert eure Anwendung entsprechend, sodass beim Abhaken einer Aufgabe diese an das Ende der Liste verschoben wird. Wird die Aufgabe anschließend erneut angeklickt und als offen markiert, so soll sie wieder an dem Anfang der Liste verschoben werden.

Aufgabe 3: Aufgabenliste leeren

Der Benutzer soll die Möglichkeit bekommen, alle Aufgaben in seiner Liste mit einem Klick zu entfernen.

- Erstellt einen zweiten Button mit dem Label “Liste leeren”!
- Implementiert die Funktionalität des Buttons, so dass beim Klick alle Aufgaben aus der Liste entfernt werden. Dabei sollen die Aufgaben nicht abgehakt, sondern gänzlich entfernt werden!

Da war es für den ersten Teil dieses Übung! Im nächsten Teil kümmern wir uns darum, dass Aufgaben dauerhaft gespeichert werden und beim Laden der Anwendung personalisiert angezeigt werden.