Modul 294

Inhalt

[XAMPP 3](#_Toc124162721)

[VS Erweiterungen 3](#_Toc124162722)

[Launch.json config 3](#_Toc124162723)

[Font herunterladen und installieren 3](#_Toc124162724)

[Icon einfügen (einfarbig) 4](#_Toc124162725)

[Farbe ändern 4](#_Toc124162726)

[Icon einfügen (zweifarbig) 4](#_Toc124162727)

# XAMPP

X

Apache

MySQL

PhP

Pearl

# VS Erweiterungen

Auto rename Tag

Handlebars Snippets

GitHub Copilot (kostenpflichtig)

jQuery Code Snippets

Materialize Snippets

# Launch.json config

{

    // Use IntelliSense to learn about possible attributes.

    // Hover to view descriptions of existing attributes.

    // For more information, visit: https://go.microsoft.com/fwlink/?linkid=830387

    "version": "0.2.0",

    "configurations": [

        {

            "type": "msedge",

            "request": "launch",

            "name": "Open with Edge",

            "url": "http://localhost:8080/${relativeFile}",

            "webRoot": "${workspaceFolder}"

        }

    ]

}

# Font herunterladen und installieren

Fonts.google.com

Font auswählen und herunterladen

<style>

        @font-face {

            font-family: "name";

            src: url(dateiname.endung) format('format');

        }

    </style>

# Icon einfügen (einfarbig)

    <span class="material-symbols-outlined">

        home

        </span>

Das in den Body

<link rel="stylesheet" href="https://fonts.googleapis.com/css2?family=Material+Symbols+Outlined:opsz,wght,FILL,GRAD@20..48,100..700,0..1,-50..200" />

Das in den head

.material-symbols-outlined {

            font-variation-settings:

                'FILL' 0,

                'wght' 400,

                'GRAD' 0,

                'opsz' 48

        }

Hier kann man das Icon noch anpassen

Codeschnipsel von: fonts.google.com/icons

## Farbe ändern

.material-symbols-outlined {

            color: #0000FF;

einfach über font-variation-settings: color: farbe schreiben

# Icon einfügen (zweifarbig)

<span class="material-icons-two-tone">

        home

        </span>

<link href="https://fonts.googleapis.com/icon?family=Material+Icons+Two+Tone" rel="stylesheet">

Ganz hinten muss die Icon sorte hinzugefügt werden, css ist aus [Leitfaden zu Material-Symbolen  |  Google Fonts  |  Google Developers](https://developers.google.com/fonts/docs/material_icons#icon_font_for_the_web)

Auf der Seite [Hex Color To CSS Filter Converter - Isotropic](https://isotropic.co/tool/hex-color-to-css-filter) erstellt man den CSS Filter und fügt ihn hier ein:

<span class="material-icons-two-tone"

        style="filter: invert(9%) sepia(98%) saturate(7030%) hue-rotate(246deg) brightness(88%) contrast(145%);">

        home

    </span>

# Css

Css ist immer relative pfad angeben

Element:focus{}: wenn der Cursor um Inputfeld ist etc

Element:nth-child(odd){}: bestimmte formatierung im element, hier alle ungeraden

Element:first-child{}:erstes element im element

Body>p{}: direkte p-elemente im body

Body p{} alle p im body

P [data-id=’’4’’]{}: p mit attribut data-id=’’4’’

# Problembehandlung

F12 -.> dev-tools

Konsole: gibt es Errors?

Netzwerk: sind alle Dateien erreichbar?

Lol

# JavaScript

## Einbinden

Am Schluss einbinden oder so im header:

<script src="index.js" defer></script>

Intern

<script>

        console.log("Hoppid");

    </script>

Extern

<script src="index.js"></script>

Im head

## Logging

Console.log(“hello”);

console.warn(“hello”);

console.error(“hello”)

## commands

document.getElementById('title').innerHTML = "hello world";

das Element mit der ID title bekkommt den Inhalt hello world

## Rechnen

a = 6;

b = "6";

c = a + b;

console.log(c);

//66

a = 6;

b = "6";

c = a - b;

console.log(c);

//0

a = 6;

b = "6";

c = a \* b;

console.log(c);

//36

a = 6;

b = "6";

c = a /  b;

console.log(c);

//1

//junge was das

+ wird verwendet um text zu vereinen, deshalb gibt die erste Rechnung 66 aus.

## Variablen

let arr1 = [1, 2, 3, 4];

//Array

let obj1 = {

    name: "John",

    age: 30

}

//Objekt

console.log(obj1.name);

//Name des objektes

console.log(typeof arr1);

//Typ der Variable

## Funktionen

function test()

{

    //dosth

}

Test.js

<h1 id="title" onclick="test">title</h1>

Test.html

let text = '<br> Das ist ein Test';

document.write(text);

gibt fehlermeldung  


muss ins HTML geschrieben warden in einen Script-tag

## forEach

arr1.forEach((value, index) => {

    console.log(value);

    console.log(index);

})

## Callback

function myDisplayer(some) {

    document.getElementById("demo").innerHTML = some;

}

function myCalculator(num1, num2, myCallback) {

    let sum = num1 + num2;

    myCallback(some);

}

myCalculator(5, 5, myDisplayer);

callback ist quasi eine Variable für eine andere Funktion

# Frameworks & Libraries

jQuery  
React.js  
D3.js  
Bootstrap  
Angular.js  
Vue.js  
Ember.js  
Node.js  
Next.js  
Ionic

# jQuery

link:

<script src="https://cdnjs.cloudflare.com/ajax/libs/jquery/3.6.3/jquery.min.js"></script>

Content ändern

<script>

        $('#title').html('hopp');

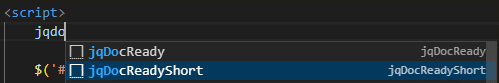
    </script>

Document.getelementbyid wird erstetz durch $(‘title’)

Element ausgeben

console.log($('#title').html);

## document ready

  
shortcut für code unten

 $(function () {

//code

        });

Führt code aus wenn das dokument geladen wurde

## jQClick



        $('#id').click(function (e) {

            e.preventDefault();

        });

Wird ausgeführt wenn element geklickt

        $('#id').click(function (e) {

            e.preventDefault();

            $(this).html('clicked'); //zeigt clicked an

            $('body').append('<hr> <h1>test</h1>'); //fügt das nach append im body hinzu

        });

## Add or remove class

$('#title').addClass('blue');

$('#title').removeClass('blue');

## Input output

<body>

    <input id="name" type="text">

    <button id="send">senden</button>

    <script>

        $('#send').click(function (e) {

            e.preventDefault();

            let name = $('#name').val();

            console.log(name);

            $('body').append('hello ' + name)

        });

    </script>

</body>

# CSS Framework

Sammlung von Gestaltungselementen und Hilfsmitteln für einfaches und standardisiertes Webdesign mit CSS.

Responsive Elemente

## Bootstrap

Beliebtestes Framework

Geht weg von jQuery

Arbeiten mit SASS

Recht langsam

Lange einarbeiten

## Materialize CSS

Weit verbreitet

Nahe am Android design

Strenge Designsprache

[Getting Started - Materialize (materializecss.com)](https://materializecss.com/getting-started.html)

    <!-- Compiled and minified CSS -->

    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/materialize/1.0.0/css/materialize.min.css">

    <!-- Compiled and minified JavaScript -->

    <script src="https://cdnjs.cloudflare.com/ajax/libs/materialize/1.0.0/js/materialize.min.js"></script>

Diese beiden Codes einbinden

### Farben

[Color - Materialize (materializecss.com)](https://materializecss.com/color.html)

Hier Farbe auswählen und als Klasse ins Element einbinden

<div class="teal accent-3 //Hintergrund

blue-text text-lighten2" //Schriftfarbe

>dinifettimuater</div>

### NavBar

[Navbar - Materialize (materializecss.com)](https://materializecss.com/navbar.html)

Links nicht klickbar

$('a').click(function (e) {

    e.preventDefault();

});

# Handlebars

<script src="https://cdnjs.cloudflare.com/ajax/libs/handlebars.js/4.7.7/handlebars.min.js"></script>

Link einbetten

Template erstellen

|  |  |
| --- | --- |
| JSON Datei  {      "data":[          {              "id": 1,              "name": "Passat",              "treibstoff": "Diesel",              "farbe": "0000FF",              "bauart": "Limousine",              "tank": "Tank"          },          {              "id": 2,              "name": "Opel",              "treibstoff": "Benzin",              "farbe": "00FF00",              "bauart": "SUV",              "tank": "Tank"          },          {              "id": 3,              "name": "VW",              "treibstoff": "Elektro",              "farbe": "FF0000",              "bauart": "Pickup",              "tank": "Tank"          }      ]  } | HBS Datei  {{#each data}}  <tr data-id="{{id}}">              <td>{{id}}</td>              <td>{{name}}</td>              <td>{{treibstoff}}</td>              <td>{{farbe}}</td>              <td>{{bauart}}</td>              <td>{{tank}}</td>              <td><i class="small material-icons blue-text btn carBtn">directions\_car</i>                  <i href="#modal1" class="small material-icons green-text btn penBtn modal-trigger">edit</i>                  <i class="small material-icons red-text btn deleteBtn">remove\_circle</i>              </td>          </tr>  {{/each}} |

Bezeichnungen (hellblau) müssen übereinstimmen

Einbinden

$.get("template/table.hbs",

    function (response) {

        //console.log(response);

        var template = Handlebars.compile(response);

        $.getJSON("data/data.json",

            function (response) {

                //console.log(response);

                $('tbody').html(template(response));

            });

    },);

Damit die Buttons in der Auto-Webseite funktionieren, müssen diese Funktionen in die

$.get("template/table.hbs",

funktion

# Materialize css ändern

Live Sass Compiler installieren

Download SASS von [Getting Started - Materialize (materializecss.com)](https://materializecss.com/getting-started.html)

SASS ordner ins Projekt

Anpassen was nötig

Compilieren

Beide dabei entstehenden Dateien in den CSS ordner

Einbinden anstelle von Materialize-Seite