Science Camp Informatik

Hallo Mädels,
schön dass ihr hier seid.
Unsere Unterlagen zusammen mit Demos der jeweiligen Aufgabe findet ihr unter
https://sciencecampinformatik.github.io/2016/

Falls ihr wärend oder auch nach dem Camp noch fragen zu HTML, CSS oder Javascript habt könnt ihr uns gerne Mails schreiben unter:

- Lena Winter: lena.winter@fsmi.uni-karlsruhe.de
- Matthias Holoch: mholoch@gmail.com

Auf den Folgenden Seiten findet ihr die Anleitungen der Workshopaufgaben, sowie cheatsheets zu HTML, CSS und Javascript.

Aufgabe: Uhr

i. Javascript anlegen und verlinken:
   - Lege eine Javascript Datei für die Uhr an.
   - Verlinke diese als skript im head deiner HTML-Seite

`<script src="..."> </src>`
2. HTML:
* Füge ein DIV mit der id "uhr" ein:

`<div id="uhr"></div>`
3. Javascript:
* Lege eine Funktion zeitanzeigen() an:

`function zeitanzeigen(){ ... }`
* Lege eine neue Variable an und speichere die aktuelle Zeit darin:

`var currentTime = new Date();`
* Lege jeweils eine Variable für die Sekunden, Minuten und Stunden an und lies die aktuellen Sekunden, Minuten und Stundne aus dem Datum aus:

`var stunden = currentTime.getHours();`

`var minuten = currentTime.getMinutes();`

`var sekunden = currentTime.getSeconds();`
4. Javascript:
* Zeige die zeit auf deiner Seite an:

`document.getElementById("uhr").textContent = stunden + ":" + minuten + ":" + sekunden;`
5. Javascript:
* Lass deine Uhr ticken: Schreibe über deine Funktion ganz oben in deine Javascriptdatei:

`setInterval("zeitanzeigen()", 1000);`
Dieser Ausdruck ruft deine Funktion zeitanzeigen jede Sekunde einmal auf.
6. Javascript:
* Wenn du führende Nullen also 08 statt nur 8 für Sekunden, Minuten und Stunden anzeigen willst füge für alles drei jeweils folgenden code ein und tausche "sekunden" entsprechend aus:

`if (sekunden < 10) { sekunden= "0" + sekunden; }`
7. CSS:
* Wenn du Lust dazu hast kannst du deine Uhr auch hübscher aussehen lassen indem du in deiner CSS-Datei ein Feld #uhr hinzufügst und reinschreibst was du möchtest:

`#uhr{ ... }`

Aufgabe: Bilder Galerie

i. Html-Wiederholung:

- Füge zwei Buttons auf deiner Internetseite hinzu einen um weiter zu schalten und einen um zurück zu schalten.

  `<button>...</button>`
  * Füge auf deiner Internetseite ein Bild ein und gibt diesem Element eine id.

  `<img id =”bild” src = “...” />`
  2. Javascript Bilder für die Galerie:
  * Für eine Bildergalerie brauchst du natürlich auch Bilder suche dir mindestens 3 Stück heraus.
  * Erstelle ein Array mit den Links zu den Bildern als Elemente.

  `var bilder = [“...”, “...”, “...”];`
  Wobei ihr in den Anführungszeichen immer den Pfad zu eurem Bild braucht. Setze das Bild, das bereits in HTML eingebunden ist, zusätzlich als erstes Element in das Array.
  3. Javascript Variablen:
  * Füge eine Variable hinzu die den Index des aktuell angezeigten Bildes enthält. Der Index ist die Position an der euer Bild im Array steht.

  `var indexaktuell = 0;`
  * Füge eine Variable hinzu, die den größten Index in deinem Array enthält. Das entspricht der Länge des Arrays minus 1.

  `var max = bilder.length -1;`
  4. Javascript Funktion `next`:
  * Erstelle eine Funktion next.

  `function next(){}`
  * Diese Funktion soll beim Aufruf das nächste Bild in der Galerie anzeigen. Also den Index des momentan angezeigten Bildes um eins erhöhen. Und dieses dann auch anzeigen

  `function next(){ indexaktuell = indexaktuell + 1; document.getElementById(“bild”).src = bilder[indexaktuell]; }`
  * Achtung: Beachte den Fall wenn der Index des angezeigten Bildes größer werden würde als der maximale Index deines Arrays. Um diesen Fall zu beachten benutzt du eine if-Abfrage:

  `function next(){ if(indexaktuell = = max){ indexaktuell = 0; }else{ indexaktuell = indexaktuell + 1; } document.getElementById(“bild”).src = bilder[indexaktuell]; }`
  * Diese Funktion soll beim click auf den Button zum weiterschalten in der Galerie aufgerufen werden
  * fügt in den Tag eures Button, der weiter schalten soll,
  `<button onclick=”next()”>` hinzu.

ii. Javascript Funktion previous:

- Diese Funktion ist sehr ähnlich zu next. Versuche das mal alleine.

Aufgabe: Geburtstagskalendar

i. HTML:
   - Dieses mal brauchen wir nicht viel HTML. Schreibe eine Überschrift und die tags für eine ungeordnete Liste.
   - In die tags für die Liste kannst du schonmal den Aufruf deiner Funktion schreiben. Diesesmal einfach in script-tags.

ii. Javascript Array:
   - Ersteinmal brauchst du Einträge für die Geburtstage die du angezeigt bekommen möchtest. Such dir min 2 Leute die im August Geburtstag haben und Bilder von ihnen.
   - Jetzt kommt das ganze in ein Array. Pro Person willst du aber mehrere Sachen Speichern deswegen benutzt du als Arrayelemente Objekte. Ein solches Objekt sieht dann so aus:

```
{
    name: Max Mustermann,
    month: 8,
    date: 1.8.2000,
    src: “irgendein Bild”
}
```

i. Javascript aktuelles Datum:
   - Du brauchst eine neue Variable in der du das aktuelle Datum speicherst. das aktuelle Datum bekommst du über den Aufruf “new Date()”.

ii. Javascript Function:
   - Lege eine neue Funktion an. Diese soll jedes Arrayelement anschauen und wenn der Monat des Eintrages der aktuelle Monat ist soll sie eine Ausgabe machen, die wir in 5. behandeln. Den aktuellen Monat kannst du über array[i].month abfragen wenn array dein Arrayname und i der Index deines Arrayelementes ist. Den aktuellen Monat bekommst du über date.getMonth(), wenn date dein Datum ist.
   - Achtung: Januar ist der Monat 0, Dezember ist 11

iii. Ausgabe HTML und Javascript:
   - Unser Ziel ist es, dass von allen Leuten die Geburtstag haben das Bild sowie Name und Geburtsdatum ausgegeben wird. Das ganze soll dann eine Liste sein, in der alle Personen untereinander aufgeführt sind.
   - Für die Ausgabe verwenden wir “document.write()”. Diese Funktion schreibt das was ihr als Parameter gegeben wird direkt an die Stelle ins html an der sie aufgerufen wird. In unserem Fall also zwischen den Listentags. Am besten schreibst du dir jetzt den HTML-Code auf für das was du am Ende als Ausgabe haben möchtest. Also ein Listenelement das ein Bild sowie einen Paragraphen beinhaltet, in dem Name und Geburtsdatum der Person steht.
   - Jetzt wird es ein bisschen kompliziert. Schau dir deinen HTML code an und füge vor allen Anführungszeichen ein \ ein. Das brauchst du weil Anführungszeichen in Javascript das Ende oder den Anfang von Strings markieren. Deshalb muss man sie “escapen” das heißt man sagt dem Computer das er sie als Anführungszeichen und nicht als Stringanfang oder Ende Lesen soll.
   - Jetzt musst du alle Informationen die du aus deinem Array bekommst makieren und dein HTML davor und dahinter auseinander ziehen und dort ein Plus einfügen zB so: `<p> + name + + datum +</p>`
   - Als nächstes fügst du vor und hinter alle zusammenhängenden HTML Stücke Anführungszeichen ein. Also sähe unser Beispiel so aus: `“<p>”+ name + ” “ + datum + “</p>”`
   - Als nächstes fügst du statt deinem Bild, name und datum die Elemente aus dem Array ein. auf diese greifst du genauso zu wie auf dem Monat aus dem Array.
     Falls du nicht weiterkommst, kannst du uns gerne fragen oder auf der Demoseite von sciencecampinformatik.github.io/2016 nachschauen.

Aufgabe: Kartenapp mit LeafletJS

Heute lernst du, wie man Code, den andere bereits mal geschrieben haben, für deine eigene Webseite verwenden und benutzen kannst. Solche Programme, die du für deine Entwicklung nutzen kannst, heißen *Library* (*Bibliothek*) oder *Framework* . Wir schauen uns heute leafletjs.com an um eine Webseite zu entwickeln.

Geh zuerst auf http://sciencecampinformatik.github.io/2016/ und schau dir auf der Map-Demo-Seite an, wie es aussehen soll.

Fremden Code zu benutzen ist meistens am Anfang sehr verwirrend. Dafür ist es umso großartiger, wenn es dann funktioniert und man plötzlich tolle, neue Funktionen hat.
Wie immer: Wenn Du irgendwo Probleme hast und nicht weiterkommst, frag uns einfach.

Erste Erfahrungen mit Leaflet JS

Leaflet ist ein sogennantes Open Source Projekt. Open Source Projekte sind Projekte, deren Code öffentlich ist und bei denen jeder, der will, mitarbeiten kann. In dieser Hinsicht ist es ein wenig wie Wikipedia, wo jeder der möchte Beiträge schreiben kann.

Bevor Du Leaflet in deiner Website benutzen kannst musst du den Leaflet-Code in deiner Website einbinden. Sonst weiß der Browser nicht, wo er den Code findet.

Binde Leaflet in deine Webseite ein

i. Die Dateien, die den Leaflet-Code enthalten befinden sich schon auf dem USB-Stick, den Du von uns bekommen hast. Du findest sie unter `bibliothek/leaflet`
ii. In diesem Ordner findest Du die Datei `leaflet.js`, die enthält allen Javascript Code von Leaflet.
iii. Außerdem gibt es ein Stylesheet `leaflet.css`, das für die Leaflet-Elemente ein Aussehen definiert.
iv. Die Datei `leaflet-src.js` brauchst Du nicht. (Sie enthält den selben Code wie `leaflet.js`, ist allerdings nicht komprimiert und daher noch lesbar)
v. Jetzt bist Du bereit Leaflet in den Header deiner Website einzubinden. Du brauchst den Javascript-Code und das CSS-Stylesheet. Das Einbinden der beiden Dateien funktioniert genau so, wie Du auch schon deine eigenen Javascript- und CSS-Dateien eingebunden hast: (es können beliebig viele Javascript- und CSS-Dateien in eine Website eingebunden werden)

```
<header>
  ...
  <!-- Binde den Leaflet-Javascript-Code ein -->
  <script src="bibliothek/leaflet/leaflet.js"/>
  <!-- Binde das Leaflet-Stylesheet ein -->
  <link rel="stylesheet" type="text/css" href="bibliothek/leaflet/leaflet.css
"/>
  ...
</header>
```

Leaflet sollte jetzt korrekt eingebunden sein und kann benutzt werden!

Jetzt brauchst Du zuerst einen Bereich auf deiner Website, in der die Karte angezeigt werden soll. Erstelle dazu in deiner HTML-Datei ein neues `div`-Element und gebe ihm die id “map”:

```
<body>
  ...
  <div id="map">
  </div>
  ...
</body>
```

In diesem Bereich wird später die Karte auftauchen!

Da der eingebundene Javascript-Code von Leaflet von sich aus überhaupt nichts tut, brauchst Du noch ein bisschen eigenen Javascript-Code.
Auf der Startseite von leafletjs.com ist ein Beispiel Code gegeben.

Probiere den Beispiel Code aus

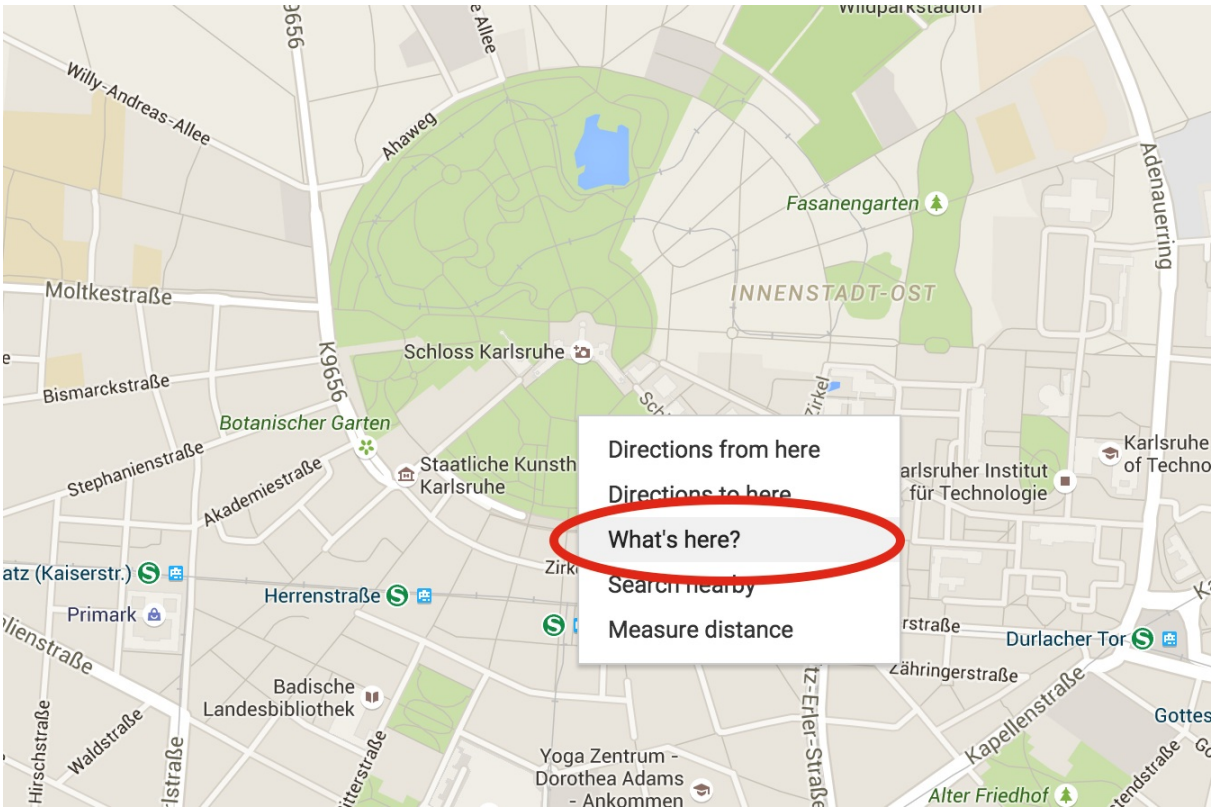
i. Erstelle eine Datei namens `map.js`, am besten im Unterordner `js`.
ii. Schreibe den Beispielcode in `map.js`.
iii. Binde `map.js` in deine Website hinter dem Karten-`div` Element ein:

```
<body>
  ...
  <div id="map">
  </div>
  <script src="js/map.js"/>
  ...
</body>
```

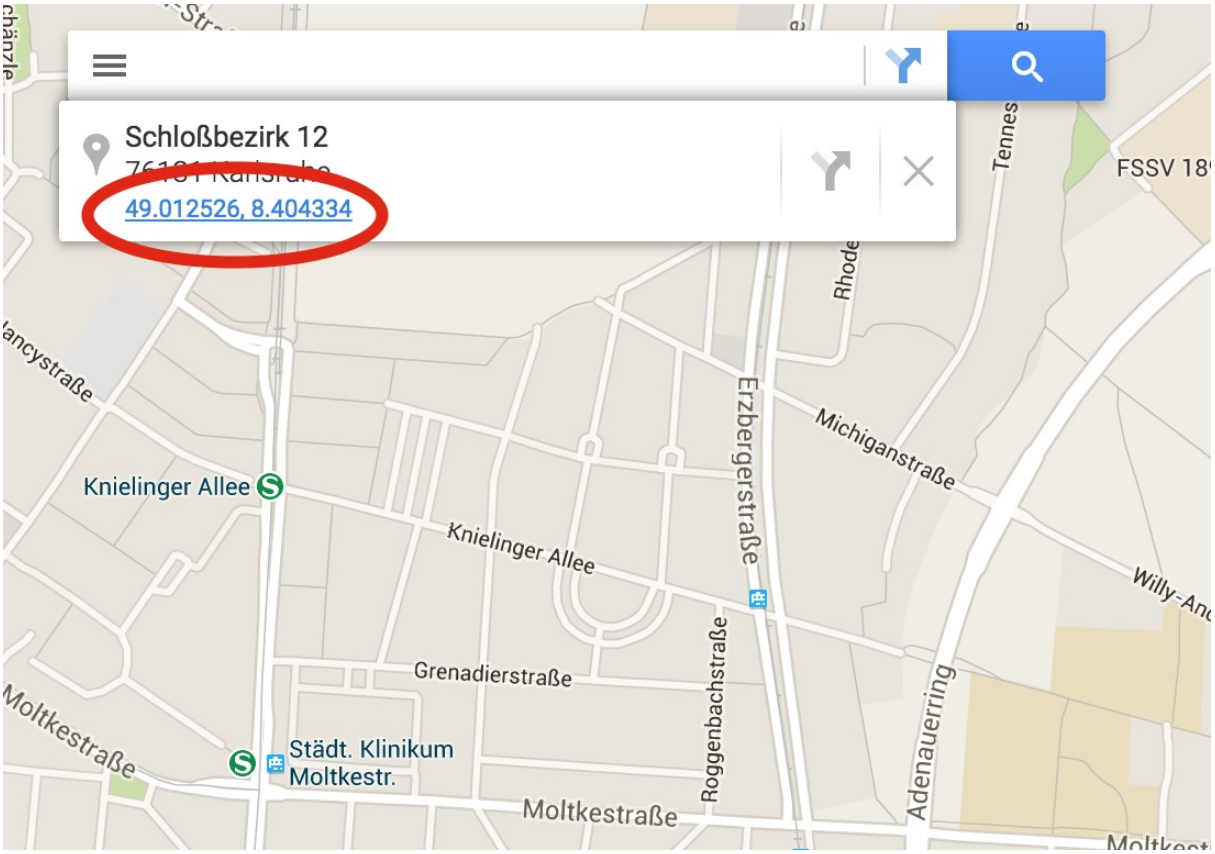
Jetzt sollte Deine Website eine Karte anzeigen!
Wenn das soweit funktioniert, wollen die Karte ein wenig anpassen.

Verändere den Code so, dass die Karte Karlsruhe anzeigt.

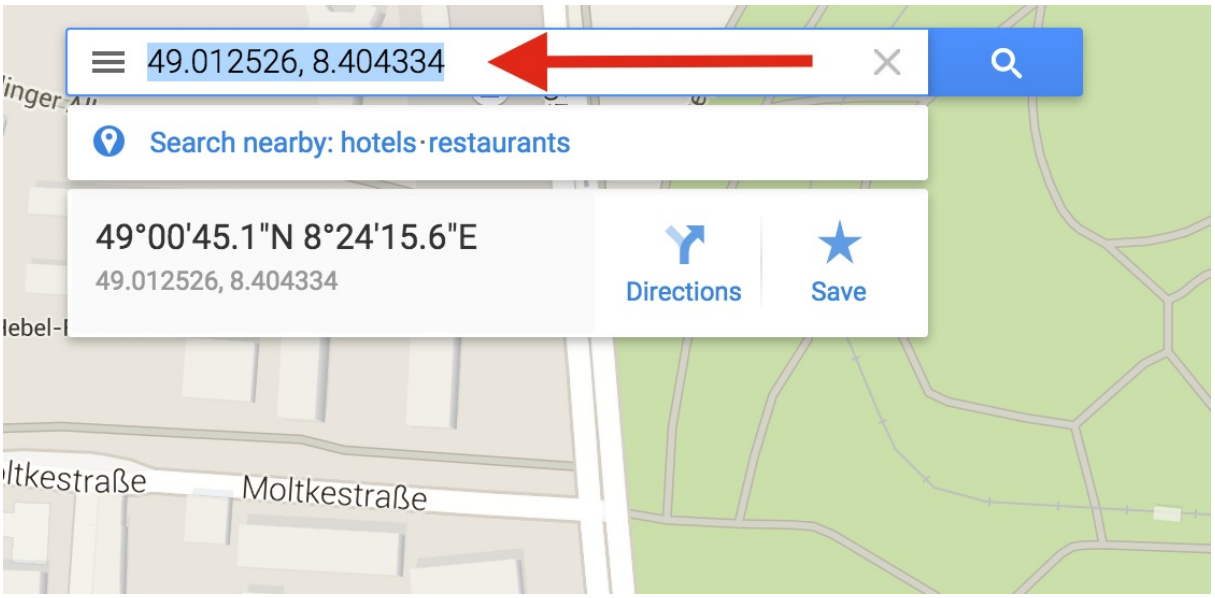
i. Lösche den Marker in London. Wir brauchen diesen nicht.
   - Schau dir den Beispielcode genau an.
   - Welcher Teil erstellt den Marker und das Popupfenster?
   - Wenn du es gefunden hast, entferne den Code.
ii. Zentriere nun die Karte auf Karlsruhe
   - Welche Funktion könnte dafür verantwortlich sein?
   - Suche in der Docs von Leaflet nach dieser Funktion und versuche zu verstehen, wie sie funktioniert. (Das ist nicht einfach zu verstehen. Frag nach Hilfe!)
iii. Um Leaflet sinnvoll zu benutzen brauchst du immer wieder Koordinaten von Orten. Du kannst dafür GoogleMaps benutzen
   - Mache einen Rechts-klick auf den Ort, dessen Koordinate du willst
   - Klicke auf *Whats' here?*



- Klicke auf die Koordinate, die nun am Suchfenster erscheint



- Kopiere dir die Koordinate



iv. Verändere nun den Code so, dass die Karte auf Karlsruhe zentriert ist.

Setze nun verschiedene Marker auf die Seite.

i. Schau dir nochmal den Code an, den wir vorhin gelöscht haben.
ii. Verwende ihn wieder im Marker auf die Karte zu setzen.
   - Falls du die gleichen Marker wie in der Demo haben möchtest, findest du hier die Koordinaten:
     - var infobau = [49.013817, 8.419790];
     - var schlosspark = [49.013149, 8.404364];
     - var klinikum = [49.015609, 8.374574];
     - var gameforge = [49.019921, 8.440989];
     - var his = [49.010194, 8.418930];

Tipp: Schreibe die Koordinaten nicht direkt in den Marker rein, sondern definiere erst Variablen mit den Koordinaten und benutze dann diese.

Navigationsangaben

Nun wollen wir auch die Route zwischen zwei Orten berechnen. Leaflet selbst bietet nicht die Möglichkeit, Routen zu berechnen. Daher benutzen wir wieder eine andere Library. Sie nennt sich *Leaflet Routing Machine*. Der Code der Library befindet sich ebenfalls bereits auf eurem USB-Stick unter `bibliothek/leaflet-routing-machine.

Binde Leaflet Routing Machine ein.

i. Gehe auf die Webseite http://www.liedman.net/leaflet-routing-machine/
ii. Suche den Abschnitt *Getting Started* und binde die Library entsprechend ein
   - Im Beispiel auf der Webseite wird zusätzlich zu Leaflet Routing Machine auf Leaflet eingebunden - da wir Leaflet bereits eingebunden haben, brauchen wir nur 2 der 4 Codezeilen. Welche sind das?
iii. Teste Leaflet Routing Machine, indem du den Beispielcode kopierst und bei dir im JavaScript-Code einfügst.
iv. Ändere den Beispielcode, sodass eine Route zwischen Markern berechnet wird, die du bereits gesetzt hast.

Verschiedene Routen anzeigen

Auf der Demoseite siehst du, dass in der 3. Version eine Route zwischen dem Infobau und einem anderen

Marker berechnet wird. Wenn du auf den Button “Nächste Route” klickst, wird eine andere Route startend vom Infobau angezeigt.

Versuche dies nachzubauen

i. Du brauchst einen Button, der eine Funktion in JavaScript aufruft.
ii. Schreibe diese Funktion, die aufgerufen wird. Lass die Funktion zunächst erstmal leer.
iii. Bestimme einen Marker, der immer dein Startpunkt sein wird.
iv. Erstelle ein Array, welches verschiedene Koordinaten hat, zu dem die Route berechnet wird.
v. Schreibe die Funktion, sodass immer ein neues Ziel gewählt wird.
vi. Erweitere die Funktion nun, sodass die Route zwischen Start und dem neuen Ziel berechnet wird.

Tipp: Ein neues Ziel im Array zu bestimmen, wenn man auf einen Button klickt, funktioniert sehr ähnlich wie in der Bildergalerie.

Sonstiges

Du kannst auch andere Kartendesigns für deine Webseite wählen. Diese Designs nennt man für Karten “Vector Tiles”. Normalerweise kosten Vector Tiles Geld, weil es sehr teuer ist, die vielen Bilder zu hosten. Einige sind allerdings kostenlos. Ihr dürft allerdings auf keinen Fall die Attribution dieser Tiles löschen.

Eine Attribution ist ein Hinweis, von wem die Tiles eigentlich kommen, die du benutzt. Außerdem wird in der Attribution erklärt unter welchen rechtlichen Bedingungen du die Vector Tiles benutzen darfst. Eine Attribution ist quasi ein fairer Tausch - du darfst die Vector Tiles kostenlos benutzen, dafür musst du aber auch angeben, von wem die Tiles kommen. Die Attribution siehst du unten rechts in deiner Karte.

Dies ist ein üblicher Vorgang in der Open Source Welt. Hier sind mögliche Tiles, die du benutzen darfst. Probiere sie aus und wähle dein Lieblingsdesign:

```
L.tileLayer('http://a.tile.openstreetmap.fr/hot/{z}/{x}/{y}.png', {
    attribution: '&copy; <a href="http://osm.org/copyright">OpenStreetMap</a>
 contributors | Tiles by <a href="http://hot.openstreetmap.org/">HOT</a>'
}).addTo(map);
```

```
L.tileLayer('http://{s}.tile.osm.org/{z}/{x}/{y}.png', {
    attribution: '&copy; <a href="http://osm.org/copyright">OpenStreetMap</a>
 contributors'
}).addTo(map);
```

```
L.tileLayer('http://otile1.mqcdn.com/tiles/1.0.0/map/{z}/{x}/{y}.jpg', {
    attribution: 'Tiles Courtesy of <a href="http://www.mapquest.com/" target
="_blank">MapQuest</a> <img src="http://developer.mapquest.com/content/osm/mq
_logo.png">, &copy; Data by <a href="http://openstreetmap.org">OpenStreetMap<
/a>, under <a href="http://www.openstreetmap.org/copyright">ODbL</a>.'
}).addTo(map);
```

```
L.tileLayer('http://c.tile.stamen.com/toner/{z}/{x}/{y}.png', {
    attribution: 'Map tiles by <a href="http://stamen.com">Stamen Design</a>,
 under <a href="http://creativecommons.org/licenses/by/3.0">CC BY 3.0</a>. Da
ta by <a href="http://openstreetmap.org">OpenStreetMap</a>, under <a href="ht
tp://www.openstreetmap.org/copyright">ODbL</a>.'
```

```
}).addTo(map);
```

```
L.tileLayer('http://c.tile.stamen.com/watercolor/{z}/{x}/{y}.png', {
    attribution: 'Map tiles by <a href="http://stamen.com">Stamen Design</a>,
 under <a href="http://creativecommons.org/licenses/by/3.0">CC BY 3.0</a>. Da
ta by <a href="http://openstreetmap.org">OpenStreetMap</a>, under <a href="ht
tp://www.openstreetmap.org/copyright">ODbL</a>.'
}).addTo(map);
```

Falls du nicht weiterkommst, kannst du uns gerne fragen oder auf der Demoseite von sciencecampinformatik.github.io/2016 nachschauen.

Cheatography

HTML5 deutsch Cheat Sheet

by Karionis (Karionis) via cheatography.com/2670/cs/683/

Struktur

Kopfbereich	<header>…</header>
Navigation	<nav>…</nav>
Sinnabschnitt, z. B. Kapitel	<section>…</section>
geschlossener Inhalt, z. B. News	<article>…</article>
semantisch neutraler Abschnitt	…
Ergänzung, z. B. Seitenspalte	<aside>…</aside>
Fußbereich	<footer>…</footer>

Navigation Beispiel

```
<nav>
  <ul>
    <li><a href="…">…</a></li>
    <li><a href="…">…</a></li>
    <li><a href="…">…</a>
      <ul>
        <li><a href="…">…</a></li>
        <li><a href="…">…</a></li>
      </ul>
    </li>
    <li><a href="…">…</a></li>
  </ul>
</nav>
```

Text

Überschrift	…
	…
	…
	…
	…
	…
Absatz	…
echte Leerzeile	
Zeilenumbruch	
Hervorhebung Fettschrift	…

Text (cont)

Hervorhebung Kursivschrift	…
Stiländerung zu Fettschrift	…
Stiländerung zu Kursivschrift	…
kurzes Zitat	<q>…</q>
langes Zitat	<blockquote>…</blockquote>
Abkürzung	<abbr>…</abbr>
vorformatierter Text	<pre>…</pre>
Programmcode	<code>…</code>
hochgestellter Text	[…]
tiefgestellter Text	_…
Inline-Abschnitt	…
horizontale Trennlinie	<hr>

Link

Link externe Seite	SPON
Link interne Seite	Impressum
Definition Sprungmarke einer Seite	z. B. <h1 id="inhalt">Inhaltsangabe</h1>
Link zur Sprungmarke der Seite	Inhaltsangabe
Link als E-Mail	

Achtung: Mailadressen sollten zum Schutz vor Spam **nicht** als Klartext angegeben werden.

Bild

```
<img src="bild.jpg" alt="Bildbeschreibung">
```

Abbildung

```
<figure>
  <img src="…">
  <figcaption>Bildunterschrift</figcaption>
</figure>
```

Tabelle

Tabellendefinition	<table>…</table>
Tabellenüberschrift	<caption>…<caption>
Tabellenkopf	<thead>…</thead>
Tabellenrumpf	<tbody>…</tbody>
Tabellenfuß	<tfoot>…</tfoot>
Einleitung Spaltendefinition	<colgroup>…</colgroup>
Spaltendefinition	<col>…</col>
Zeile	<tr>…</tr>
Spaltenkopf	<th>…</th>
Spaltenzelle	<td>…</td>

By **Karionis** (Karionis)
cheatography.com/karionis/

Published 29th November, 2012.
Last updated 30th November, 2012.

Cheatography

HTML5 deutsch Cheat Sheet

by Karionis (Karionis) via cheatography.com/2670/cs/683/

Tabelle Beispiel

```
<table border="1">
  <caption>Tagesablauf</caption>
  <tr>
    <th>Uhrzeit</th>
    <th>Programm</th>
  </tr>
  <tr>
    <td>09:00 bis 13:00 Uhr</td>
    <td>Kinderprogramm</td>
  </tr>
  <tr>
    <td>13:00 bis 14:30 Uhr</td>
    <td>Mittagessen</td>
  </tr>
  <tr>
    <td>14:30 bis 17:30 Uhr</td>
    <td>Familienprogramm</td>
  </tr>
</table>
```

Liste

geordnete Liste	…
ungeordnete Liste	…
Listeneintrag geordn./ ungeordn. Liste	…
Definitionsliste	<dl>… </dl>
Definitionsl. Ausdruck	<dt>… </dt>
Definitionsl. Erläuterung	<dd>… </dd>

Formular

Formulardefinition	<form action="…" method="…">…</form>
Abschnittsdefinition	<fieldset>…</fieldset>
Erläuterung des Abschnitts	<legend>…</legend>
Eingabefeldbezeichner	<label for="labelbezeichner">…</label>
Eingabefeld einzeiliger Text	<input type="text" name="variable" id="labelbezeichner">
Eingabefeld mehrzeiliger Text	<textarea>…</textarea>
Eingabefeld Passwort	<input type="password" …>
Eingabefeld Kontrollfeld	<input type="checkbox" …>
Eingabefeld Optionsfeld	<input type="radio" …>
Eingabefeld Bild	<input type="image" src="bild.jpg" …>
Eingabefeld Schalter „Absenden“	<input type="submit" …>
Auswahlliste	<select>…</select>
Auswahlliste Eintrag	<option>…</option>
Auswahlliste Eintraggruppe	<optgroup>…</optgroup>

Formular Beispiel

```
<form method="post"
action="abonnement.php">
  <fieldset>
    <legend>Newsletter-
Abonnement</legend>
    <label for="e-mail-
adresse">E-Mail</label><input type="email"
name="email" id="e-mail-adresse"><br>
    <input type="submit">
  </fieldset>
</form>
```

Kommentar HTML

```
<!-- Kommtentartext -->
```

CSS2.1 deutsch Cheat Sheet

by Karionis (Karionis) via cheatography.com/2670/cs/685/

Abmessungen

height:	auto;	Höhe (Standard)
	100px;	
	20%;	
	5.0em;	
min-height:	(siehe height)	minimale Höhe
max-height:	(siehe height)	maximale Höhe
width:	(siehe height)	Breite
min-width:	(siehe height)	minimale Breite
max-width:	(siehe height)	maximale Breite

Positionierung

position:	static;	Standard
	relative;	relativ zur ursprünglichen Position
	absolute;	abhängig vom Elternelement
	fixed;	abhängig vom Browserfenster
top:	auto;	Standard
bottom:	100px;	
left:	20%;	

Positionierung (cont)

right:	-2.0em;

top, *bottom*, *left*, *right* werden nur bei *position: relative*, *absolute* und *fixed* beachtet.

Floating

float:	none;	Standard
	left;	Element steht links und wird rechts umflossen
	right:	Element steht rechts und wird links umflossen.
clear:	none;	Standard
	left;	Aufheben des Umfließens links
	right;	Aufheben des Umfließens links
	both;	Aufheben beidseitigen Umfließens

Farbe

color:	red;	Vordergrundfarbe (Text)
	#789ABC;	hexadezimaler Farbwert
	rgb(0,25,50);	RGB-Farbwerte
background-color:	(siehe color)	Hintergrundfarbe

Kommentar

/* ***Kommentartext*** */

Textformatierung

font-family:	"Times New Roman"	Schriftart (Times)
	Arial, Helvetica, sans-serif;	serifenlose Schrift
font-size:	12pt;	Schriftgröße (Punkt)
	10px;	Pixel
	1.0em;	relativ zum Elternelement
	120%;	relativ zur Grundgröße
font-style:	normal;	Schriftstil (Standard)
	italic;	kursiv
	oblique;	schräg
font-weight:	lighter;	Schriftstärke (dünn)
	normal;	Standard
	bold;	
	bolder;	
line-height:	normal;	Zeilenhöhe (Standard)
	14pt	Punkt
	20px	Pixel
	120%	relativ zur Schriftgröße
	1.2em;	relativ zum Elternelement

By **Karionis** (Karionis)
cheatography.com/karionis/

Published 29th November, 2012.
Last updated 29th November, 2012.

Cheatography

CSS2.1 deutsch Cheat Sheet

by Karionis (Karionis) via cheatography.com/2670/cs/685/

Textformatierung (cont)

letter-spacing:	normal;	Zeichenabstand (Standard)
	0.05em;	relativ zum Elternelement
text-align:	left;	Ausrichtung (Standard)
	right;	
	center;	zentriert
	justify;	Blocksatz
text-decoration:	none;	Schriftauszeichnung (Standard)
	underline;	unterstrichen
	line-through;	durchgestrichen
text-indent:	1.0em;	Einrückung
	10%;	
text-transform:	none;	Textauszeichnung (Standard)
	capitalize;	Kapitälchen
	uppercase;	Großbuchstaben
	lowercase;	Kleinbuchststaben

Innenabstand

padding:	0;	keiner
	2px 4px;	2 Pixel oben u. unten, 4 Pixel rechts u. links
	1pt 2px 3% 4.0em;	Abstand oben, rechts, unten, links
padding-top:	10px;	Abstand oben
padding-bottom:	(s.o.)	Abstand unten
padding-left:	(s.o.)	Abstand links
padding-right:	(s.o.)	Abstand rechts

Aussenabstand

margin:	0;	keiner
	10px 20px;	(s. padding)
	1pt 2px 3% -4.0em;	(s. padding)
margin-top:	10px;	Abstand oben
margin-bottom:	(s. o.)	Abstand unten
margin-left:	(s. o.)	Abstand links
margin-right:	(s. o.)	Abstand rechts

Außenabstände dürfen auch negative Werte annehmen.

Rahmen

border:	1px solid;	Rahmen 1 Pixel Stärke
	thin dotted red;	dünner, roter und gepunkteter Rahmen
	0.2em dashed #D0D0D0;	gestrichelter grauer Rahmen
border-top:	(s. o.)	oberer Rahmen
border-bottom-:	(s. o.)	unterer Rahmen
border-left:	(s. o.)	linker Rahmen
border-right:	(s. o.)	rechter Rahmen
border-weight:	1px 2px 3px 4px;	Rahmenstärke
border-style:	solid;	Rahmenstil
border-color:	black gray silver white;	Rahmenfarbe

By **Karionis** (Karionis)
cheatography.com/karionis/

Published 29th November, 2012.
Last updated 29th November, 2012.

JavaScript Cheat Sheet

Variablen

Um eine Variable mit dem Namen `nummer` anzulegen, musst du folgendes schreiben:

```
var nummer;
```

Danach kannst du dieser Variablen Werte hinzufügen, in dem du diese mit `=` zuweist:

```
nummer = 3;
```

Du kannst diese zum Beispiel auch hoch zählen

```
nummer = nummer + 1;
```

oder eine andere Funktion aufrufen:

```
nummer = prompt("Gib eine Nummer ein.", "");
```

Vergesst vor allem nicht die Semikolons ;

If-Else Bedingungen

Mit `if` kannst du Bedingungen festlegen. Du kannst damit Befehle umsetzen wie: “Wenn die Nummer gleich 0 ist, dann möchte ich die Nummer um eine Zahl erhöhen.” Der Code dazu wäre dafür:

```
if (nummer == 0) {
    nummer = nummer + 1;
}
```

Etwas allgemeiner ausgedrückt, werden If-Bedingungen also so geschrieben:

```
if (BEDINGUNG) {
    BEFEHLE, falls BEDINGUNG wahr ist;
}
```

Man kann auch weitere Fälle abfragen. Wenn du zum Beispiel die Nummer unterschiedlich erhöhen möchtest, je nachdem, was Nummer ist, geht das so:

```
if (nummer == 0) {
    nummer = nummer + 1;

} else if (nummer == 1) {
    nummer = nummer + 2;

} else if (nummer == 2) {
    nummer = nummer + 3;
}
```

Falls du z.B. die Nummer erhöhen möchtest, wenn sie 0 ist, und sie in allen anderen Fällen eins abziehen möchtest, kannst du `else` benutzen.

```
if (nummer == 0) {
    nummer = nummer + 1;
} else {
    nummer = nummer - 1;
}
```

Beschreiben der Bedingungen

Du kannst die Bedingungen unterschiedlich beschreiben:

- `==` , wenn etwas *gleich* sein soll
- `!=` , wenn etwas *ungleich* sein soll
- `<=` , wenn etwas *kleiner oder gleich* sein soll
- `>=` , wenn etwas *größer oder gleich* sein soll
- `>` , wenn etwas *größer* sein soll
- `<` , wenn etwas *kleiner* sein soll

Außerdem kannst du auch mehrere Bedingungen abfragen:

- `&&` bedeutet *und*
- `||` bedeutet *oder*

Beispiele:

- Wenn die Nummer **größer** als 4 **oder gleich** 0 sein soll:

  ```
  if (nummer > 4 || nummer == 0) {
      ...
  }
  ```

- Wenn die Nummer **nicht** 1 **und** auch **nicht** 2 sein soll:
  `if (nummer != 1 && nummer != 2) { ... }`

Schleifen

Wenn du einen Befehl hast, der sehr oft aufgerufen werden soll, kannst du Schleifen benutzen.

```
for (var counter = 0; counter < 100; counter = counter + 1) {
    BEFEHL
}
```

In diesem Fall wird der `BEFEHL` 100x ausgeführt.

Als Erklärung, was genau passiert:

1. Wir setzen einen Counter auf 0 (`var counter = 0`).
2. Wir erhöhen den Counter um 1 (`counter = counter + 1`).
3. Wir führen den `BEFEHL` aus.
4. Wir schauen nach, ob der Counter kleiner 100 ist (`counter < 100`)
    - falls ja, wiederholen wir Schritt 2-4
    - falls nein, hören wir auf

Anmerkung: `counter = counter + 1` macht übrigens genau das gleich wie `counter++` .

Funktionen

Funktionen sind Codeabschnitte oder zusammengefasste Befehle. Wenn du gewisse Befehle hast, die du immer wieder ausführen willst, oder wenn du Befehle hast, die du über einen Button ausführen möchtest, benutzt du Funktionen:

```
function showName() {
    document.getElementById("name").textContent = "Lena";
}
```

Dabei ist `showName` der **Name** der Funktion. Der Inhalt oder die Befehle der Funktion sind in den geschweiften Klammern `{ }` .

Wenn du nun `showName()` irgendwo aufrufst, dann werden alle Befehle der Funktion ausgeführt.

Arrays

Du kannst dir Arrays vorstellen wie einen Schubladenschrank:

```
var zahlen = [7, 2, 4, 5];
```

Die Variable `zahlen` ist nun ein Schubladenschrank.

- In der ersten Schublade ist die Zahl 7.
- In der zweiten Schublade ist die Zahl 2.
- In der dritten Schublade ist die Zahl 4.
- In der vierten Schublade ist die Zahl 5.

```
var fruechte = ["apfel", "banane", "birne"];
```

Wäre entsprechend ein Array mit den Wörtern `apfel`, `banane` und `birne`.

Wenn ihr nun aus dem Schubladenschrank, also aus dem Array ein Element rausnehmen wollt, geht das mit eckigen Klammern:

```
var frucht = fruechte[1];
```

Würde bedeuten, dass ihr aus dem Array `fruechte` das 1. Elemente rausnehmt und in die Variable `frucht` reinschreibt. In `frucht` steht also nun `banane` drin.

Ihr könnt auch Sachen in die Schublade reintun bzw. Sachen in das Array schreiben:

```
zahlen[2] = 3;
```

Damit schreibt ihr in die 2. Schublade von Zahlen den Wert 3. Und das Array verändert sich von:

- [7, 2, 4, 5] zu
- [7, 2, 3, 5]

Vergesst nicht, dass bei Arrays das Zählen bei 0 anfängt.

JavaScript in HTML

Datei einbinden

Um eine Funktion aus einer JavaScript-Datei in deiner HTML-Datei zu benutzen, muss die HTML-Datei irgendwoher wissen, wo diese JavaScript-Datei liegt. Das macht ihr mit diesem Code:

```
<script src="PFAD_ZUR_JAVASCRIPT_DATEI"></script>
```

Wenn ihr eine JavaScript-Datei habt, die z.B. `code.js` heißt und im gleichen Ordner liegt, schreibt ihr:

```
<script src="code.js"></script>
```

Wenn die JavaScript-Datei im Unterordner `js` liegt, schreibt ihr:

```
<script src="js/code.js">
```

JavaScript-Funktion aus Datei einbinden

Wenn ihr mit einem Button eine JavaScript-Funktion aufrufen wollt, könnt ihr das so machen:

```
<button onclick="FUNKTIONSNAME()">Drück mich</button>
```

Als Beispiel: Wenn ihr eine Funktion, die `showName()` heißt und ungefähr so aussieht:

```
function showName() {
    name = prompt("Gib deinen Namen ein", "");
}
```

dann müsst ihr im Button folgendes aufrufen:

```
<button onclick="showName()">Drück mich</button>
```